

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日      2 0 0 3 年   8 月 2 8 日  
Date of Application:

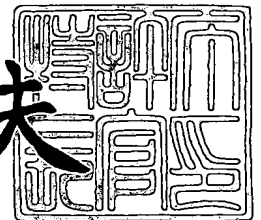
出 願 番 号      特 願 2 0 0 3 - 3 0 5 5 1 3  
Application Number:  
[ST. 10/C] :      [ J P 2 0 0 3 - 3 0 5 5 1 3 ]

出 願 人      株式会社リコー  
Applicant(s):

2 0 0 3 年   9 月 3 0 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康 夫



【書類名】 特許願  
【整理番号】 0306366  
【提出日】 平成15年 8月28日  
【あて先】 特許庁長官 殿  
【国際特許分類】 G03G 21/00 396  
【発明者】  
    【住所又は居所】 東京都大田区中馬込 1 丁目 3 番 6 号 株式会社リコー内  
    【氏名】 松島 弘幸  
【特許出願人】  
    【識別番号】 000006747  
    【住所又は居所】 東京都大田区中馬込 1 丁目 3 番 6 号  
    【氏名又は名称】 株式会社リコー  
    【代表者】 桜井 正光  
【代理人】  
    【識別番号】 100080931  
    【住所又は居所】 東京都豊島区東池袋 1 丁目 2 0 番 2 号 池袋ホワイトハウスビル  
                    8 1 8 号  
    【弁理士】  
    【氏名又は名称】 大澤 敬  
【先の出願に基づく優先権主張】  
    【出願番号】 特願2002-276451  
    【出願日】 平成14年 9月24日  
【先の出願に基づく優先権主張】  
    【出願番号】 特願2002-272978  
    【出願日】 平成14年 9月19日  
【手数料の表示】  
    【予納台帳番号】 014498  
    【納付金額】 21,000円  
【提出物件の目録】  
    【物件名】 特許請求の範囲 1  
    【物件名】 明細書 1  
    【物件名】 図面 1  
    【物件名】 要約書 1  
    【包括委任状番号】 9809113

**【書類名】 特許請求の範囲****【請求項 1】**

第 1 の通信装置が、第 2 の通信装置に送信すべき動作要求と前記第 2 の通信装置から受信した動作要求に対する動作応答とを一括して前記第 2 の通信装置に送信し、

前記第 2 の通信装置が、前記第 1 の通信装置に送信すべき動作要求と前記第 1 の通信装置から受信した動作要求に対する動作応答とを一括して前記第 1 の通信装置に送信することを特徴とする通信方法。

**【請求項 2】**

請求項 1 記載の通信方法であって、

前記動作要求は関数呼び出しであり、

前記動作応答はその関数呼び出しによって呼び出された関数の実行結果であることを特徴とする通信方法。

**【請求項 3】**

請求項 1 又は 2 記載の通信方法であって、

前記動作要求及び前記動作応答の送信は、常に前記第 2 の通信装置から通信要求を発しを行い、前記第 1 の通信装置からの前記動作要求及び前記動作応答の送信は、前記第 2 の通信装置からの通信要求に対する通信応答として行うことを特徴とする通信方法。

**【請求項 4】**

請求項 3 記載の通信方法であって、

前記第 2 の通信装置が前記第 1 の通信装置に対して定期的に通信要求を行うことを特徴とする通信方法。

**【請求項 5】**

第 1 の通信装置が、第 2 の通信装置に送信すべき SOAP リクエストと前記第 2 の通信装置から受信した SOAP リクエストに対する SOAP レスポンスとを 1 つのメッセージに記載して前記第 2 の通信装置に送信し、

前記第 2 の通信装置が、前記第 1 の通信装置に送信すべき SOAP リクエストと前記第 1 の通信装置から受信した SOAP リクエストに対する SOAP レスポンスとを 1 つのメッセージに記載して前記第 1 の通信装置に送信することを特徴とする通信方法。

**【請求項 6】**

請求項 5 記載の通信方法であって、

前記 SOAP リクエストには関数呼び出しを記載し、

前記 SOAP レスポンスはその関数呼び出しによって呼び出された関数の実行結果を記載することを特徴とする通信方法。

**【請求項 7】**

請求項 5 又は 6 記載の通信方法であって、

前記第 2 の通信装置は、前記第 1 の通信装置に送信すべき SOAP リクエスト及び SOAP レスポンスを、常に HTTP リクエストに記載して送信し、

前記第 1 の通信装置は、前記第 2 の通信装置に送信すべき SOAP リクエスト及び SOAP レスポンスを、常に前記 HTTP リクエストに対する HTTP レスポンスに記載して送信することを特徴とする通信方法。

**【請求項 8】**

請求項 7 記載の通信方法であって、

前記第 2 の通信装置が前記第 1 の通信装置に対して定期的に HTTP リクエストを送信することを特徴とする通信方法。

【書類名】 明細書

【発明の名称】 通信方法

【技術分野】

【0001】

この発明は、複数の通信装置の間で、動作要求及び通信相手から受信した動作要求に対する動作応答の送受信を行う通信方法に関する。

【背景技術】

【0002】

従来から、通信装置をネットワークを介して接続した通信システムにおいて、通信装置同士で互いにメッセージを交換させることにより、通信相手の装置に対して通知や要求を行わせることが行われている。そして、このようなシステムにおいて、ある装置から別の装置に動作要求としてコマンドを送信して動作を実行させ、送信相手から動作の実行結果を動作応答として返信させることも行われている。

【0003】

このような技術は、例えば特許文献1に開示されており、この文献には、リモートプロセッサがローカルプロセッサに対して実行されるべきコマンドを指示するメッセージを送信し、そのコマンドに対する応答を受信することが記載されている。

また、この文献には、ローカルプロセッサがファイアウォールの内側に配置されている場合において、ローカルプロセッサからファイアウォールの外側のリモートプロセッサに通信要求を送信し、リモートプロセッサがこの通信要求に対する応答としてローカルプロセッサに対してコマンドを送信するようにすることにより、ファイアウォールの外側から内側に向けてコマンドを送信できるようにする技術も開示されている。

【特許文献1】 特開2001-273211号公報

【0004】

また、このような動作要求に関する技術は、通信装置に接続された装置の動作を遠隔制御するシステムにも適用することができる。特許文献2には、ブラインド及び照明を操作する機能を有する遠隔被操作装置に、ユーザからの操作を受け付ける機能を有する遠隔操作装置からコマンドを送信してブラインド及び照明を操作させる遠隔操作システムにこのような技術を適用した例が記載されている。ただし、この文献には、コマンドに対する応答を送信する点は示されていない。

【特許文献2】 特開2002-135858号公報

【発明の開示】

【発明が解決しようとする課題】

【0005】

ところで、複数の通信装置間でメッセージを交換する場合において、コマンドを送信する通信装置は、1つとは限らない。複数の通信装置が互いに相手に対してコマンドを送信するようにすることも可能であり、この場合には、コマンドを受け付けた通信装置に、それぞれコマンドの送信元に対して実行結果を返させるようにすることが求められている。そして、このような動作を行う場合、ある通信装置から通信相手の装置に送信する情報としては、通信相手の装置に対するコマンドと、通信相手の装置から受信したコマンドについての実行結果とが考えられる。

【0006】

従来は、これらのコマンドと実行結果とは別々に送信するようにしていた。しかし、このような方式では、コマンドの送信時と受信したコマンドに対する実行結果の送信時とに、それぞれ別々に通信のコネクションを確立する必要がある。従って、通信のオーバーヘッドが大きくなり、効率性の点で問題があった。

現状では、ネットワークを介した通信をダイヤルアップ接続で行う環境もまだ多く残っており、このような環境においては上記の点が特に問題となる。このような環境では、コネクションの確立に数十秒単位の時間を要することもあり、またコネクションを確立する毎に料金を課金されるので、コネクションを確立する回数が増加するとコストアップにつ

なかるためである。

この発明は、このような問題を解決し、複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する場合において、通信の効率を上げることを目的とする。

【課題を解決するための手段】

【0007】

上記の目的を達成するため、この発明の通信方法は、第1の通信装置と第2の通信装置とが通信する場合において、第1の通信装置が、第2の通信装置に送信すべき動作要求と上記第2の通信装置から受信した動作要求に対する動作応答とを一括して上記第2の通信装置に送信し、上記第2の通信装置が、上記第1の通信装置に送信すべき動作要求と上記第1の通信装置から受信した動作要求に対する動作応答とを一括して上記第1の通信装置に送信するようにしたものである。

【0008】

このような通信方法において、上記動作要求を関数呼び出しとし、上記動作応答を、その関数呼び出しによって呼び出された関数の実行結果とするとよい。

さらに、上記動作要求及び上記動作応答の送信を、常に上記第2の通信装置から通信要求を発して行い、上記第1の通信装置からの上記動作要求及び上記動作応答の送信は、上記第2の通信装置からの通信要求に対する通信応答として行うようにするとよい。

さらにまた、上記第2の通信装置が上記第1の通信装置に対して定期的に通信要求を行うようにするとよい。

【0009】

また、この発明は、第1の通信装置が、第2の通信装置に送信すべきSOAPリクエストと上記第2の通信装置から受信したSOAPリクエストに対するSOAPレスポンスとを1つのメッセージに記載して上記第2の通信装置に送信し、上記第2の通信装置が、上記第1の通信装置に送信すべきSOAPリクエストと上記第1の通信装置から受信したSOAPリクエストに対するSOAPレスポンスとを1つのメッセージに記載して上記第1の通信装置に送信する通信方法も提供する。

【0010】

このような通信方法において、上記SOAPリクエストには関数呼び出しを記載し、上記SOAPレスポンスはその関数呼び出しによって呼び出された関数の実行結果を記載するようにするとよい。

さらに、上記第2の通信装置が、上記第1の通信装置に送信すべきSOAPリクエスト及びSOAPレスポンスを、常にHTTPリクエストに記載して送信し、上記第1の通信装置が、上記第2の通信装置に送信すべきSOAPリクエスト及びSOAPレスポンスを、常に上記HTTPリクエストに対するHTTPレスポンスに記載して送信するようにするとよい。

さらにまた、上記第2の通信装置が上記第1の通信装置に対して定期的にHTTPリクエストを送信するようにするとよい。

【発明の効果】

【0011】

以上のようなこの発明の通信方法によれば、通信相手に対する動作要求と、通信相手から受信した動作要求に対する動作応答とを、一度コネクションを確立するのみで送信することができるので、オーバーヘッドを低減し、通信の効率を上げることができる。

【発明を実施するための最良の形態】

【0012】

以下、この発明を実施するための最良の形態について、図面を参照して説明する。

まず図1に、この発明を適用する通信システムの構成例を示す。

この通信システムは、図1に示すように、第1の通信装置1と第2の通信装置2とをネットワーク10によって接続して構成している。

そして、第1の通信装置1及び第2の通信装置2は、通信機能を備えたPC等のコンピ

ユータを始め、通信機能及び情報処理機能を備えた各種電子装置として構成することができる。ネットワーク10としては、インターネットやLAN（ローカルエリアネットワーク）を始め、有線、無線を問わず、ネットワーク通信が可能な各種通信経路を用いることができる。

#### 【0013】

また、第1の通信装置1及び第2の通信装置2は、互いの制御管理を行うためのアプリケーションプログラムを実装している。そして、これらの各ノードは、RPC（Remote Procedure Call）により、互いの実装するアプリケーションプログラムのメソッドに対する処理の依頼である「動作要求」を送信し、この依頼された処理の結果である「動作応答」を取得することができるようになっている。即ち、第1の通信装置1は、第2の通信装置2への要求（以下、第1の通信装置側要求という）を生成してこれを第2の通信装置2へ引き渡し、この要求に対する応答を取得できる一方で、第2の通信装置2は、第1の通信装置1への要求（以下、第2の通信装置側要求という）を生成してこれを第1の通信装置1へ引き渡し、この要求に対する応答を取得できるようになっている。

なお、ここではメソッドを入力と出力の形式を規定した論理的な関数として定義するものとする。そしてこの場合、動作要求はこの関数を呼び出す関数呼び出し（Procedure Call）となり、動作応答はその関数呼び出しによって呼び出された関数の実行結果となる。

#### 【0014】

図2に、これらの動作要求と動作応答の関係を示す。

図2（A）は、第1の通信装置1で第2の通信装置2に対する動作要求が発生したケースである。このケースでは、第1の通信装置1が第1の通信装置側動作要求を生成して第2の通信装置2に送信し、これを受け取った第2の通信装置2がその要求に対する動作応答を返すというモデルになる。

#### 【0015】

図2（B）は、第2の通信装置2で第1の通信装置1に対する動作要求が発生したケースである。このケースでは、第2の通信装置2が第2の通信装置側要求を生成して第1の通信装置1に送信し、これを受け取った第1の通信装置1がその要求に対する動作応答を返すというモデルになる。

なお、ここではRPCによる引数並びに戻り値の受け渡しのプロトコルとしてSOAP（Simple Object Access Protocol）を採用し、上記の動作要求や動作応答は、ここではSOAPメッセージとして記載するようにしている。

#### 【0016】

この発明の特徴は、このように複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する場合において、通信相手の装置に送信すべき動作要求とその通信相手の装置から受信した動作要求に対する動作応答とを一括して送信するようにする点である。

そして、実際に動作要求や動作応答を転送するための通信プロトコルとしては、システムの構成に合わせて適当なものを採用することができ、例えばHTTP（HyperText Transfer Protocol）やSMTP（Simple Mail Transfer Protocol）を採用することができる。

そこで、この発明の通信方法に関して、まず通信プロトコルにHTTPを採用する場合の実施例について説明し、次にSMTPを採用する場合の実施例について説明する。

#### 【0017】

〔HTTPを採用する場合の実施例：図3乃至図21〕

図3に、HTTPを採用する場合の実施例を適用する通信システムの構成例を示す。

この通信システムは、図3に示すように、HTTPサーバ12とHTTPクライアント11とをインターネット13によって接続して構成している。ただし、セキュリティを向上させるため、HTTPクライアント11はファイアウォール14を介してインターネット13に接続するようにしている。そして、HTTPサーバ12が第1の通信装置に、HTTPクライアント11が第2の通信装置に該当する。

**【0018】**

なお、HTTPを用いて通信を行う場合、ファイアウォール14の内側にあるノードに対しては、ファイアウォール14の外側からは自由にアクセスできず、そのノードからの通信要求(HTTPリクエスト)に対する通信応答(HTTPレスポンス)という形でしかデータを送信できない。そこで、この通信システムにおいては、ファイアウォール14の内側にあるノードをHTTPクライアント11、外側にあるノードをHTTPサーバ12としているのである。従って、これらの各ノードの機能は、これら相互間の通信以外においては、クライアントあるいはサーバに限定する必要はない。

**【0019】**

また、HTTPサーバ12及びHTTPクライアント11は、図1に示した第1の通信装置1及び第2の通信装置2の場合と同様に、互いの制御管理を行うためのアプリケーションプログラムを実装している。そして、RPC(Remote Procedure Call)により、互いの実装するアプリケーションプログラムのメソッドに対する処理の依頼である「動作要求」を送信し、この依頼された処理の結果である「動作応答」を取得することができるようになっている。

**【0020】**

図4に、これらの動作要求と動作応答の関係を示す。

図4(A)は、HTTPクライアント11でHTTPサーバ12に対する動作要求が発生したケースである。このケースでは、HTTPクライアント11がクライアント側動作要求(以下、「クライアントコマンド」とも呼ぶ)を生成してHTTPサーバ12に送信し、これを受け取ったHTTPサーバ12がそのコマンドに対する動作応答(以下、「コマンド応答」あるいは単に「応答」とも呼ぶ)を返すというモデルになる。

**【0021】**

図4(B)は、HTTPサーバ12でHTTPクライアント11に対する動作要求が発生したケースである。このケースでは、HTTPサーバ12がサーバ側動作要求(以下、サーバコマンドとも呼ぶ)を生成してHTTPクライアント11に送信し、これを受け取ったHTTPクライアント11がそのコマンドに対する動作応答を返すというモデルになる。

このように、動作要求及び動作応答は、RPCのレベルではHTTPクライアント11とHTTPサーバ12との間で対称に取り扱われるものである。しかし、通信のレベルでは対称ではない。

**【0022】**

図5にこの通信システムにおける通信シーケンスの例を示す。この通信シーケンスは、この発明の通信方法の実施形態を示すものである。

この図に示すように、この通信システムにおいては、通信は常に、HTTPクライアント11から通信要求としてHTTPリクエストをHTTPサーバ12に送信し、HTTPサーバ12からこの通信要求に対する通信応答としてHTTPレスポンスをHTTPクライアント11に返すという手順で行われる。例えばHTTPクライアント11が送信したHTTPリクエストXに対してHTTPサーバ12がHTTPレスポンスXを返し、同じくHTTPリクエストYに対してHTTPレスポンスYを返すという具合である。

**【0023】**

そして、HTTPリクエストには、HTTPクライアント11からHTTPサーバ12に送信する動作要求であるクライアントコマンドと、HTTPサーバ12からHTTPクライアント11に送信されてきたサーバコマンドに対する応答(コマンド応答)とを記載して送信するようにしている。また、HTTPレスポンスには、HTTPサーバ12からHTTPクライアント11に送信する動作要求であるサーバコマンドと、HTTPクライアント11からHTTPサーバ12に送信されてきたクライアントコマンドに対する応答(コマンド応答)とを記載して送信するようにしている。

**【0024】**

従って、例えばクライアントコマンドAは、HTTPリクエストXに記載して転送し、

コマンド応答をそのHTTPリクエストXと対応するHTTPレスポンスXに記載して転送することができる。しかし、サーバコマンドCについては、HTTPリクエストXと対応するHTTPレスポンスXに記載して転送し、そのコマンド応答は次のHTTPリクエストであるHTTPリクエストYに記載して転送することになる。

#### 【0025】

また、上記図4 (A) のケースでは、クライアントコマンドが生成された後直ちにHTTPクライアント11がHTTPサーバ12と接続を確立し、HTTPリクエストにこれを含めて引き渡すことができるが、上記図4 (B) のケースでは、HTTPクライアント11側に設置されたファイアウォール14がHTTPサーバ12からのHTTPリクエストを遮断するため、HTTPサーバ12側からHTTPクライアント11へアクセスしてサーバコマンドを直ちに引き渡すことができない。

#### 【0026】

なお、クライアントコマンド及びサーバコマンドに対する応答をそれぞれ任意の数ずつ (0でもよい) 1つのHTTPリクエストに記載することができ、サーバコマンド及びクライアントコマンドに対する応答をそれぞれ任意の数ずつ (0でもよい) 1つのHTTPレスポンスに記載することができる。そして、1つのHTTPリクエスト又はHTTPレスポンスに記載した内容は、論理的に一括して転送する。

そして、このようにすることにより、必要な情報を転送するために必要な接続の回数を減らし、オーバーヘッドを低減して通信の効率化を図っている。

#### 【0027】

図6にこの通信システムにおける別の通信シーケンスの例を示す。この通信シーケンスも、この発明の通信方法の実施形態を示すものである。

説明のため、図5には極めて単純なシーケンス例を示したが、図6には、各HTTPリクエストやHTTPレスポンスに記載するコマンドやコマンド応答の数が一定でない例を示している。

また、コマンドを受信した場合に、次の送信機会の時点で応答を返す必要もない。例えば、図6に示すクライアントコマンドBのように、コマンドを記載したHTTPリクエストX' に対応するHTTPレスポンスX' に記載して応答を返さず、後のHTTPレスポンスY' に記載して応答を返すようにしてもよい。

もちろんサーバコマンドについても同様であり、サーバコマンドを記載したHTTPレスポンスの次のHTTPリクエストにそのコマンドに対する応答を記載する必要はない。そして、さらに後のHTTPリクエストに記載して転送すればよい。

#### 【0028】

ところで、各コマンド及びコマンド応答は、それぞれ独立して生成され、また処理に供されるべきものであるから、上記のような一括転送を行うためには、転送前にこれらのコマンドやコマンド応答を結合し、また転送後に分離する処理が必要となる。次に、HTTPクライアント11及びHTTPサーバ12のハードウェア構成と共に、このような処理を行うための機能構成及びその処理の手順について説明する。

#### 【0029】

図7は、HTTPクライアント11及びHTTPサーバ12のハードウェア構成例を示す図である。

この図に示すように、HTTPクライアント11及びHTTPサーバ12はそれぞれ、CPU31、ROM32、RAM33、SD (Secure Digital) カード34、ネットワークインタフェースカード (NIC) 35を備え、これらがシステムバス36で接続されている。

#### 【0030】

これら構成要素を更に具体的に説明すると、まずCPU31は、ROM32に格納している制御プログラムによってHTTPクライアント11又はHTTPサーバ12全体を統括的に制御する制御手段である。そして、ROM32は、CPU31が使用する制御プログラムを含む各種固定データを格納している読み出し専用メモリである。



RAM 33は、CPU 31がデータ処理を行う際のワークメモリ等として使用する一時記憶用メモリである。SDカード34は、装置の電源がオフになっても記憶内容を保持するようになっている不揮発性メモリである。NIC 35は、インターネット13を始めとするネットワークを介して通信相手と情報の送受信を行うための通信手段である。

#### 【0031】

図8は、HTTPクライアント11の機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す機能ブロック図である。

図8に示す機能のうち、クライアントコマンドプール41及びサーバコマンドプール42は、いずれかの書き換え可能な記憶手段に設けられるものである。例えばSDカード34に設けることができるが、RAM 33や図示しないHDD（ハードディスクドライブ）に設けてもよい。クライアントコマンド生成手段43、サーバコマンド実行結果生成手段44、送信メッセージ収集手段45、受信メッセージ分配手段48の機能は、CPU 31によって実現されるものである。また、HTTPリクエスト送信手段46及びHTTPレスポンス受信手段47の機能は、CPU 31及びNIC 35によって実現されるものである。

#### 【0032】

これらの機能についてさらに詳述する。

まず、クライアントコマンドプール41は、クライアントコマンドと、このコマンドに対する応答と、このコマンドの識別情報とを関連付けて登録するプールである。また、サーバコマンドプール42は、サーバコマンドと、このコマンドに対する応答と、このコマンドの識別情報とを関連付けて登録するプールである。

クライアントコマンド生成手段43は、クライアントコマンドを生成し、このコマンドを識別する識別情報（ID）を割り当て、さらにこのコマンドを管理するための管理情報を付し、これらの情報を関連付けてテーブル形式のクライアントコマンドシートとしてクライアントコマンドプール41に登録する機能を有する。このうち、クライアントコマンドを生成する部分には、例えばHTTPクライアント11に備えるアプリケーションが該当する。また、クライアントコマンド生成手段43に、HTTPサーバ12に各コマンドを実行させる際の優先順位を、生成したクライアントコマンドに付する機能を設けてもよい。

#### 【0033】

ここで、図9にクライアントコマンドシートにおけるデータ構造の例を示す。

この図に示すように、クライアントコマンドシートには、「コマンドID」、「メソッド名」、「入力パラメータ」、「状態」、「クライアントコマンド実行結果の通知先」、および「出力パラメータ」のデータを記憶する領域を設けている。そして、このうち「コマンドID」、「メソッド名」、および「入力パラメータ」がクライアントコマンド（及びそこに付されたID）に該当し、「状態」及び「クライアントコマンド実行結果の通知先」が管理情報に該当する。「出力パラメータ」は、HTTPサーバ12から受信するコマンド応答の内容である。

#### 【0034】

次に、各項目の内容について説明する。

まず、「メソッド名」は、HTTPサーバ12に対するリクエストの内容であり、HTTPサーバ12において呼び出す関数の種類を示す。「入力パラメータ」は、「メソッド名」に付随するデータであり、関数を呼び出す際の引数である。「コマンドID」は、クライアントコマンドを識別するための識別情報である。「状態」は、クライアントコマンドに関する処理の進行状況を示すデータであり、処理の進行と共に、「未送信」→「応答待ち」→「応答受信済」と遷移していく。

#### 【0035】

「クライアントコマンド実行結果の通知先」は、そのシートに記載しているクライアントコマンドに対する応答を受信した場合に、その旨を通知して必要な処理を実行させるモジュールを示す参照情報である。参照するモジュールは、クライアントコマンドを生成し

たアプリケーションであることが多いが、必ずしもそうである必要はない。「出力パラメータ」には、コマンド応答を受け取った段階で、その内容を格納する。HTTPサーバ12からのコマンド応答を受け取るまでは空である。

#### 【0036】

図8の説明に戻ると、サーバコマンド実行結果生成手段44は、サーバコマンドプール42からサーバコマンドを読み出して実行するアプリケーションである。そして、サーバコマンドに対する応答を生成し、サーバコマンドのコマンドIDと関連付けてサーバコマンドプール42に登録する機能を有する。なお、HTTPサーバ12から受信したサーバコマンドは、このコマンドを識別するID及びこのコマンドを管理するための管理情報と関連付けて、テーブル形式のサーバコマンドシートとしてサーバコマンドプール42に登録しておくようにしている。そして、サーバコマンド実行結果生成手段44が生成したコマンド応答も、実行したサーバコマンドについてのサーバコマンドシートに登録する。

#### 【0037】

また、サーバコマンド実行結果生成手段44に、サーバコマンドプール42から複数の種類のサーバコマンドを読み出し、各サーバコマンドに対する応答を生成する機能を設けることが考えられる。さらに、サーバコマンドがHTTPクライアント11に優先して処理を実行させるための実行優先順位の情報を含む場合には、優先順位の高いものから優先的に読み出して実行する機能を設けることも考えられる。

なお、サーバコマンド実行結果生成手段44は、アプリケーションそのものではなく、サーバコマンドの実行に必要なアプリケーションを呼び出してコマンドを実行させるモジュールであってもよい。

#### 【0038】

ここで、図10にサーバコマンドシートにおけるデータ構造の例を示す。

この図に示すように、サーバコマンドシートには、「コマンドID」、「メソッド名」、「入力パラメータ」、「状態」、「出力パラメータ」、および「サーバコマンドの通知先」のデータを記憶する領域を設けている。そして、このうち「コマンドID」、「メソッド名」、および「入力パラメータ」がサーバコマンド（及びそこに付されたID）に該当し、「状態」及び「サーバコマンドの通知先」が管理情報に該当する。「出力パラメータ」は、サーバコマンドの実行結果であり、HTTPクライアント11が返すコマンド応答の内容となる。

#### 【0039】

次に、各データの内容について説明する。

まず、「メソッド名」は、HTTPクライアント11に対するリクエストの内容であり、HTTPクライアント11において呼び出す関数の種類を示す。「入力パラメータ」は、「メソッド名」に付随するデータであり、関数を呼び出す際の引数である。「コマンドID」は、サーバコマンドを識別するための識別情報である。「状態」は、サーバコマンドに関する処理の状態を示すデータであり、処理の進行と共に、「未処理」→「処理完了」→「応答済」、あるいは「未処理」→「処理中」→「処理完了」→「応答済」と遷移していく。「出力パラメータ」には、サーバコマンド実行結果生成手段44によって生成された応答が格納される。サーバコマンドの実行が終了し、上記の「状態」が「処理完了」となるまでは空である。「サーバコマンドの通知先」は、サーバコマンドの実行を行うモジュールを示す参照情報である。

#### 【0040】

再び図8の説明に戻ると、送信メッセージ収集手段45は、サーバコマンド実行結果生成手段44が生成したコマンド応答とこのコマンド応答に対応するサーバコマンドのコマンドIDとを関連付けてサーバコマンドプール42から読み出すと共に、クライアントコマンド生成手段43が生成したクライアントコマンドとこのコマンドのコマンドIDとを関連付けてクライアントコマンドプール41から読み出し、これらから送信メッセージを生成する機能を有する。

なお、コマンド応答やクライアントコマンドに実行優先順位が指定されている場合には

、送信メッセージ収集手段 45 がそれぞれ実行優先順位の高いものから順に読み出すようにすることが考えられる。

#### 【0041】

ここで、送信メッセージとは、上記のコマンド応答やコマンドとコマンド ID とを、構造化言語である XML (Extensible Markup Language) で、SOAP メッセージとして記載したものである。そして、送信メッセージ収集手段 45 は、1 つのコマンド応答あるいはコマンドにつき、送信メッセージとして 1 つの SOAP メッセージを生成する。またこのとき、各コマンドのコマンド ID は SOAP ヘッダに記載し、コマンド応答及びクライアントコマンドの内容は SOAP ボディに記載する。SOAP による通信では、SOAP ヘッダと SOAP ボディとからなる SOAP エンベロープ (封筒) と呼ばれるメッセージを XML で記載し、HTTP などのプロトコルで交換することになる。

このようなコマンドやコマンド応答からの SOAP メッセージの生成は、WSDL (Web Service Description Language) に基づいて生成される所要の変換プログラム (シリアライザ) を実行し、データを直列化することによって行うことができる。

#### 【0042】

そして、HTTP リクエスト送信手段 46 は、送信メッセージ収集手段 45 が生成した送信メッセージを含む HTTP リクエストを生成し、HTTP サーバ 12 に送信する機能を有する。このとき、1 つの HTTP リクエストに送信メッセージをいくつ含めてもよいし、コマンド応答に係る送信メッセージとクライアントコマンドに係る送信メッセージとを任意に混在させることもできる。

そこで、HTTP リクエスト送信手段 46 は、これらのいずれに係る送信メッセージかに関わり無く、送信メッセージ収集手段 45 が生成した全ての送信メッセージを 1 つの HTTP リクエストに含めて送信するようにしている。ただし、1 つの HTTP リクエストに含める送信メッセージの数に上限を設けることも考えられる。

#### 【0043】

ところで、この HTTP リクエストの送信は、送信メッセージ収集手段 45 がクライアントコマンドやコマンド応答等の読み出しを試みた場合には、読み出すデータがなく、結果的に送信すべき SOAP メッセージを生成しなかった場合にも行うものである。そして、この読み出しの試みは、定期的に行うものとする。例えば、タイマによって 60 分毎に読み出すことが考えられる。

このようにするのは、上述のように、HTTP サーバ 12 から HTTP クライアント 11 に送信したい情報があっても HTTP クライアント 11 から通信を要求しない限り送信できないためである。HTTP クライアント 11 から何も送信するデータがなかったとしても、定期的に HTTP サーバ 12 に対して通信要求を送信して、HTTP サーバ 12 から HTTP クライアント 11 に情報を送信する機会を与えることにより、転送の必要な情報が長期間に亘って HTTP サーバ 12 に滞留してしまうことを防止できる。

#### 【0044】

なお、送信メッセージ収集手段 45 による読み出しと、それに続く HTTP リクエスト送信手段 46 による HTTP リクエストの送信とを、定期的なタイミング以外に適宜行ってよいことはもちろんである。例えば、緊急に送信が必要な情報がいずれかのプールに登録された場合に、クライアントコマンド生成手段 43 あるいはサーバコマンド実行結果生成手段 44 が送信メッセージ収集手段 45 にその旨を通知して読み出しを行わせるようにしてもよい。

#### 【0045】

次に、HTTP レスポンス受信手段 47 は、HTTP サーバ 12 から HTTP レスポンスを受信する機能を有する。そしてここでは、HTTP レスポンスには、サーバコマンド及びそのコマンドと関連付けられたコマンド ID を含む受信メッセージと、クライアントコマンドに対する応答及びそのコマンドと関連付けられたコマンド ID を含む受信メッセージとが、任意に混在して含まれている。

ここで、受信メッセージとは、上記のコマンドや応答とコマンド ID とを SOAP メッ

ページとして記載したものである。

#### 【0046】

受信メッセージ分配手段48は、HTTPレスポンス受信手段47が受信したHTTPレスポンスに含まれるデータを、クライアントコマンドプール41及びサーバコマンドプール42に振り分けて登録する機能を有する。

具体的には、サーバコマンド及びそのコマンドと関連付けられたコマンドIDとをサーバコマンドプール42にサーバコマンドシートを設けて登録すると共に、クライアントコマンドに対する応答については、そのコマンドと関連付けられたコマンドIDをクライアントコマンドプール41に記憶しているクライアントコマンドシートのコマンドIDと照合して対応するクライアントコマンドを特定し、そのクライアントコマンドについての「出力パラメータ」として登録する。

そしてこのとき、HTTPレスポンスを分割してそこに含まれる各受信メッセージを取り出し、そのデータをテーブルへの登録に必要な形式に変換するが、この変換は、WSDLに基づいて生成される所要の変換プログラム（デシリアライザ）を実行することによって行うことができる。

#### 【0047】

次に、このような機能を有するHTTPクライアント11がHTTPサーバ12に送信するHTTPリクエストの例を図11に示す。

このHTTPリクエストは、図11に示すように、ボディ部としてMIME (Multipurpose Internet Mail Extension) に従ったマルチパートのメッセージが記載され、この各パートには、それぞれエンティティヘッダが記載されると共に、詳細な図示は省略しているが、SOAPエンベロープが埋め込まれている。図11の例では、HTTPリクエストのHTTPボディには、「MIME\_boundary」で区分された各要素が、独立した第1パート、第2パート、第3パート、第4パートを構成しているが、HTTPボディに含めることのできるパート数は4つに限られない。0個を含め、いくつでもよい。

HTTPリクエストに埋め込まれて引き渡されるSOAPエンベロープには、クライアントコマンドを記載したものと、サーバコマンドに対する応答を記載したものとがある。

#### 【0048】

また、このような機能を有するHTTPクライアント11がHTTPサーバ12から受信するHTTPレスポンスの例を図12に示す。

図12に示すように、このHTTPレスポンスは、形式としては図11に示したHTTPリクエストとHTTPヘッダ部が異なるのみであり、ボディ部にはHTTPリクエストの場合と同様に、詳細な図示は省略しているが、MIMEに従ったマルチパートのSOAPエンベロープが記載される。SOAPエンベロープの内容については、当然コマンドやコマンド応答の内容に従って異なるものである。

HTTPレスポンスに埋め込まれて引き渡されるSOAPエンベロープには、サーバコマンドを記載したものと、クライアントコマンドに対する応答を記載したものとがある。

#### 【0049】

次に、これらのHTTPリクエスト又はHTTPレスポンスに記載されるパートの具体例を図13乃至図16に示す。

図13に示すのは、クライアントコマンドを記載したパートの例である。

この例においては、まず、エンティティヘッダの部分の「X-SOAP-Type」ヘッダに、このパートに記載されているSOAPメッセージがSOAPリクエストであるかSOAPレスポンスであるかを示す情報を記載している。この例では、値の「Request」により、SOAPリクエストであること、すなわちコマンドを記載したSOAPメッセージであることを示している。

また、「SOAPAction」ヘッダは、SOAPリクエストの内容を示すものであり、この例では、「http://www.…」というURI (Uniform Resource Identifier) によりリクエストの内容を示している。なお、「SOAPAction」ヘッダは、SOAPメッセージがSOAPレスポンスである場合には付加しないため、メッセージの受信側において、このヘッダの

有無により、SOAPメッセージがSOAPリクエストであるかSOAPレスポンスであるかを判断することもできる。

#### 【0050】

そして、「Envelope」タグの属性として、名前空間の宣言を行っている。そしてここでは、SOAPで標準として定義されている名前空間の他に、「http://www.foo.com/header」及び「http://www.foo.com/server」のURIで特定される名前空間の宣言を行っている。従って、「n」の名前空間接頭辞が付されたXMLタグについては「http://www.foo.com/header」のURIで特定される名前空間に属するタグであることがわかり、「ns」の名前空間接頭辞が付されたXMLタグについては「http://www.foo.com/server」のURIで特定される名前空間に属するタグであることがわかる。

#### 【0051】

またSOAPヘッダには、「要求ID」のXMLタグの内容として、このクライアントコマンドのIDである「12345」が記載されている。そして、SOAPボディには、クライアントコマンドシートの「メソッド名」に記憶されていたメソッドを指定する情報として、「異常通知」タグが記載され、その下位のタグ「エラーID」や「説明」の要素として、「入力パラメータ」に記憶されていた引数が記載されている。ここでは異常通知の通知内容が記載されている。

#### 【0052】

図14に示すのは、クライアントコマンドに対する応答を記載したパートの例である。この例においては、まず、エンティティヘッダの部分の「X-SOAP-Type」ヘッダの値を「Response」と記載することにより、このパートに記載されているSOAPメッセージがSOAPレスポンスであること、すなわちコマンド応答を記載したSOAPメッセージであることを示している。

また、この例においても、名前空間の宣言は図13に示した例と同様である。そして、SOAPヘッダには、「コマンドID」のXMLタグの内容として、応答を生成したクライアントコマンドのIDである「12345」が記述されている。SOAPボディには、「異常通知」コマンドに対する応答であることを示すための「異常通知Response」タグが設けられ、その下位のタグに、コマンド応答の内容が記載される。ここでは、異常通知を正常に受信した旨の情報が記載されている。そして、この情報がクライアントコマンドシートの「出力パラメータ」の項目に格納される。

#### 【0053】

図15に示すのは、サーバコマンドを記載したパートの例である。

この例においても、図13の場合と同様に、「X-SOAP-Type」ヘッダの値の「Request」により、このパートに記載されているSOAPエンベロープがSOAPリクエストであることを示し、「SOAPAction」ヘッダの情報により、SOAPリクエストの内容を示している。

#### 【0054】

また、「Envelope」タグの属性として、名前空間の宣言を行っている点も、図13の場合と同様である。そしてここでは、SOAPで標準として定義されている名前空間の他に、「http://www.foo.com/header」及び「http://www.foo.com/client」のURIで特定される名前空間の宣言を行っている。

SOAPヘッダには、「要求ID」のXMLタグの内容として、このクライアントコマンドのIDである「98765」が記載されている。そして、SOAPボディには、サーバコマンドシートの「メソッド名」に記憶されるべきメソッドを指定する情報として、「温度センサ値取得」タグが記載され、その下位のタグ「センサID」の要素として、「入力パラメータ」に記憶されるべき引数が記載されている。ここではセンサ値を取得するセンサのIDが記載されている。

なお、サーバがこのようなコマンドを送信する場合としては、例えば、クライアントからの異常通知を受けて異常の原因を特定しようとする場合等が考えられる。

#### 【0055】

図16に示すのは、サーバコマンドに対する応答を記載したパートの例である。

この例においても、図14の場合と同様に、エンティティヘッダの部分の「X-SOAP-Type」ヘッダの値を「Response」と記載することにより、このパートに記載されているSOAPメッセージがSOAPレスポンスであることを示している。

また、この例においても、名前空間の宣言は図15に示した例と同様である。そして、SOAPヘッダには、「コマンドID」のXMLタグの内容として、応答を生成したサーバコマンドのIDである「98765」が記述されている。SOAPボディには、「温度センサ値取得」コマンドに対する応答であることを示すための「温度センサ値取得Response」タグが設けられ、その下位のタグに、コマンド応答の内容が記載される。ここでは、値取得を要求されたセンサの示す温度値の情報が記載されている。

#### 【0056】

次に、以上説明したような構成及び機能を有するHTTPクライアント11において実行する処理について、図17乃至図21のフローチャートを用いて説明する。これらのフローチャートに示す処理は、HTTPクライアント11のCPU31が所要の制御プログラムを実行することによって行うものである。

#### 【0057】

まず、図17にメッセージの収集及び分配処理の基本動作のフローチャートを示す。

HTTPクライアント11のCPU31は、送信メッセージ収集手段45がクライアントコマンドやコマンド応答等の読み出しを試みるタイミングになると、図17のフローチャートに示す処理を開始する。

そして、まずクライアントコマンドの収集処理を行う(S11)。この処理は、クライアントコマンドプール41からHTTPサーバ12に送信すべきクライアントコマンドを収集する処理であり、収集したデータからSOAPエンベロープによるパートを生成する処理を含む。

#### 【0058】

次に、サーバコマンドに対する応答であるサーバコマンド実行結果の収集処理を行う(S12)。この処理は、サーバコマンドプールからHTTPサーバ12に送信すべきコマンド応答を収集する処理であり、やはり収集したデータからSOAPエンベロープによるパートを生成する処理を含む。

その後、ステップS11及びS12の処理で生成したパートを1つにマージして、すべてのパートを含むHTTPリクエストを生成し(S13)、そのHTTPリクエストをHTTPサーバ12に送信する(S14)。

ここまでの処理において、ステップS11及びS12ではCPU31は送信メッセージ収集手段45として機能し、ステップS13及びS14ではHTTPリクエスト送信手段46として機能する。

#### 【0059】

次に、HTTPリクエストに対する通信応答としてHTTPサーバ12からHTTPレスポンスを受信する(S15)。そして、受信したHTTPレスポンスのHTTPボディを各パートに分割する(S16)。ここで、各パートへの分割は、「MIME\_boundary」で区分された要素に分割することであり、またここで全てのパートに関して分割する。

そしてその後、分割して得た全てのパートを順に対象として、ステップS17乃至S19の処理を繰り返す。この処理においては、まず対象のパートがサーバコマンドを記載したパートか否か判断する(S17)。そして、サーバコマンドであればサーバコマンド登録処理を行う(S18)。また、サーバコマンドでないときは、クライアントコマンドに対する応答が記載されたパートであるので、応答通知処理を行う(S19)。

#### 【0060】

ステップS18又はS19の後は、ステップS17に戻り、次のパートを対象として処理を繰り返す。そして、全てのパートについてこれらの処理を行った時点で、図17のフローチャートに示す処理を終了する。

ここまでの処理において、ステップS15及びS16ではCPU31はHTTPレスポ

ンス受信手段47として機能し、ステップS17乃至S19では受信メッセージ分配手段48として機能する。

#### 【0061】

次に、図17のフローチャートに示した処理について、一部分ずつより詳細に示したフローチャートを用いて説明する。

図18は、図17のステップS11乃至S14の部分の処理をより詳細に示したフローチャートである。

#### 【0062】

この処理においては、HTTPクライアント11のCPU31はまず、クライアントコマンドプール41から、「状態」が「未送信」であるクライアントコマンドシートの「メソッド名」と「入力パラメータ」の内容を、送信すべきクライアントコマンドとして収集し、「コマンドID」の内容もそのコマンドのコマンドIDとして収集する(S21)。「未送信」という「状態」は、コマンドがクライアントコマンド生成手段43によって生成された後、まだHTTPサーバ12に通知されていないことを示すものであるので、これを基準にHTTPサーバ12に送信すべきコマンドを抽出できる。

#### 【0063】

その後、ステップS21で収集した全てのクライアントコマンドを順次対象として、ステップS22乃至S24の処理を繰り返す。これらの処理においては、まず対象のクライアントコマンドとそのコマンドIDとを、これらの情報がそれぞれSOAPボディとSOAPヘッダとに含まれるXML文書に変換し(S22)、対象のコマンドに関するパートとなるSOAPエンベロープを生成する(S23)。そして、対象のクライアントコマンドを記載していたクライアントコマンドシートの「状態」を「応答待ち」に変更する(S24)。「応答待ち」という「状態」は、コマンドをHTTPサーバ12に通知済であることを示すものである。

#### 【0064】

これらが全て完了した後、CPU31は、サーバコマンドプール42から、「状態」が「処理完了」であるサーバコマンドシートの「出力パラメータ」の内容を、サーバコマンドに対するコマンド応答のうち送信すべきものとして収集し、「コマンドID」の内容も、対応するサーバコマンドのコマンドIDとして収集する(S25)。「処理完了」という「状態」は、サーバコマンドに対応する処理がサーバコマンド実行結果生成手段44によって生成された後、まだHTTPサーバ12に通知されていないことを示すものであるので、これを基準にHTTPサーバ12に送信すべきコマンド応答を抽出できる。

#### 【0065】

その後、ステップS25で収集した全てのコマンド応答を順次対象として、ステップS26乃至S28の処理を繰り返す。これらの処理は、まず対象のコマンド応答とその応答と共に収集したコマンドIDとを、これらの情報がそれぞれSOAPボディとSOAPヘッダとに含まれるXML文書に変換し(S26)、対象のコマンド応答に関するパートとなるSOAPエンベロープを生成する(S27)処理である。これらの処理は、対象が異なる点以外はステップS22及びS23の処理と同じものである。そして、次に対象のコマンド応答を記載していたサーバコマンドシートの「状態」を「応答済」に変更する(S28)。「応答済」という「状態」は、コマンド応答をHTTPサーバ12に通知済であることを示すものである。

#### 【0066】

そして、ここまでの処理が全て完了した後、CPU31は、ステップS23又はS27で生成した各パートをマージし、図11に示したようなマルチパートのHTTPリクエストを生成してHTTPサーバ12に送信する(S29)。

なお、ステップS24又はS28で行った「状態」の変更は、実際にこの送信が終了してから行うようにしてもよい。このようにすることにより、通信エラーが発生しても、送信しようとしていたコマンド及びコマンド応答を再度送信の対象とすることができるので、システムの信頼性が向上する。

以上でHTTPリクエストの送信に関する処理を終了し、図17のステップS15以降に相当する処理に進む。

#### 【0067】

図19は、図17のステップS15以下の部分の処理をより詳細に示すフローチャートである。図18のステップS29の次の処理は、この図ではステップS31に該当する。

この処理においては、HTTPクライアント11のCPU31はまず、送信したHTTPリクエストに対するHTTPレスポンスの受信を待ち、HTTPサーバ12からこれを受信する(S31)。これを受信すると、そのHTTPボディを解析して各パートに分割する(S32)。

そしてその後、分割して得た各パートを順次対象として、ステップS33乃至ステップS39の処理を繰り返す。

#### 【0068】

この部分の処理においては、まず、対象のパートがサーバコマンドであるか否か判断する(S33)。上述したように、HTTPレスポンスには、クライアントコマンドに対する応答と、サーバコマンドとが含まれている可能性があるので、対象のパートがこのいずれであるかを判断するのである。そして、この判断は、対象のパートにSOAPActionヘッダが存在するか否か、あるいはX-SOAP-Typeヘッダの内容によって判断することができる。

#### 【0069】

ステップS33でサーバコマンドでなければ、そのパートはクライアントコマンドに対する応答であるので、そのパートのXML文書を解析してクライアントコマンドシートに登録できる形式のデータに変換し(S34)、クライアントコマンドプール41からそのコマンド応答に対応するクライアントコマンドを探索し、そのクライアントコマンドについてのクライアントコマンドシートの「出力パラメータ」の項目にコマンド応答のデータを登録する(S35)。なお、コマンド応答には、「コマンドID」の情報として、クライアントコマンドの送信時に付したものと同一コマンドIDが付してあるものとし、クライアントコマンドの探索は、この情報をキーとして行うことができる。

#### 【0070】

データの登録が終わると、データを登録したクライアントコマンドシートの「状態」を「応答受信済」に変更してその旨を示す(S36)。そして、「クライアントコマンド実行結果の通知先」に登録されている通知先に、応答があった旨を通知する(S37)。この通知によって、クライアントコマンドを生成したアプリケーション等は、その生成したコマンドに応答があったことを認識し、応答に応じた処理を行うことができる。

#### 【0071】

例えば、異常通知を発するアプリケーションがHTTPサーバ12に異常通知を行う旨のクライアントコマンドを生成した場合、このコマンドがHTTPサーバ12に送信されると、HTTPサーバ12はこれを正しく受け取った旨のコマンド応答を返してくる。そして、HTTPクライアント11側では、このコマンド応答を受信すると、ここに含まれるコマンドIDを基にどのクライアントコマンドに対する応答であるかを探索し、見つかったクライアントコマンドと対応させてそのコマンド応答を登録する。そして、そのコマンドの実行結果通知先として登録されている、異常通知を発するアプリケーションに、応答があった旨を通知するのである。アプリケーションは、この通知を受けた場合にクライアントコマンドシートを参照すれば、生成したコマンドの実行結果を「出力パラメータ」の項目から取得することができる。

以上のステップS37までの処理が終了すると、次のパートがあればそれを対象としてステップS33からの処理を繰り返す。

#### 【0072】

一方、ステップS33でサーバコマンドであれば、そのパートのXML文書を解析してサーバコマンドシートに登録できる形式のデータに変換し(S38)、そのサーバコマンドに対応するサーバコマンドシートを作成して、コマンドIDと共にサーバコマンドプールに登録する(S39)。ここで、サーバコマンドの内容はサーバコマンドシートの「メ



ソッド名」及び「入力パラメータ」の項目に登録し、パートに記載されていたコマンドIDは「コマンドID」の項目に登録する。また、「サーバコマンドの通知先」の項目には、「メソッド名」に記憶させたメソッドを実行させるアプリケーション等への参照情報を、予め用意してあるメソッドとアプリケーション等との対応関係の情報を参照して登録する。「状態」の初期値は「未処理」であり、「出力パラメータ」の初期値はNULLである。

#### 【0073】

以上のステップS39までの処理が終了すると、次のパートがあればそれを対象としてステップS33からの処理を繰り返す。

全てのパートについてステップS33乃至S39の処理が終了すると、図19のフローチャートに示した処理は終了する。

以上のような処理を行うことにより、HTTPクライアント11が、HTTPサーバ12に送信すべき動作要求とHTTPサーバ12から受信した動作要求に対する動作応答とを一括してHTTPサーバ12に送信することができる。また、HTTPサーバ12からの動作要求とHTTPサーバ12に送信した動作要求に対する動作応答とを一括してHTTPサーバ12から受信して処理することができる。

#### 【0074】

なお、ここでは送信すべき全てのパートを全て生成してからマージして送信を行うようにし、また全てのパートを受信してからこれを各パートに分割して処理を行うように説明したが、このようにする必要はない。

送信については、まず始めにHTTPヘッダを送信し、以後パートを生成するたびにそのパートを順次送信し、全てのパートの送信が完了した時点でその旨のデータを送信するようにしてもよい。このようにしても、これらの課程で送信されるデータが1つのみのHTTPヘッダを持つ論理的に連続した1つのHTTPリクエストであれば、1回のセッションで転送でき、ネゴシエーションの処理は1回で済むので、マージして送信する場合と同様な効果を得ることができる。また、送信すべきデータのバッファに必要なメモリ容量を低減できるので、低コストの通信装置で大きなデータを取り扱うことができる。

また、受信側でも、各パートに関する処理を、各パートを受信するたびに順次行うようにすることができる。このようにした場合に容量を低減できることは、送信側の場合と同様である。

#### 【0075】

次に、サーバコマンドの実行に関する処理について説明する。

図20は、この処理の一例を示すフローチャートである。

サーバコマンドの実行に関する処理としては、まず、図20のフローチャートに示す処理を、図19に示したステップS39の処理の後に、すなわちサーバコマンドをサーバコマンドプール42に登録した直後に行うことが考えられる。この処理において、HTTPクライアント11のCPU31は、サーバコマンド実行結果生成手段44として機能する。

#### 【0076】

そして、この処理においては、まず登録したサーバコマンドについてのサーバコマンドシートの「サーバコマンドの通知先」の情報に基づいてアプリケーション等呼び出し、「メソッド名」や「入力パラメータ」のデータを渡してサーバコマンドに関する処理を実行させる(S41)。なお、サーバコマンドに関する処理は、このフローチャートには示していないが、CPU31が別途実行することになる。

そして、これが完了すると、実行結果をサーバコマンドシートの「出力パラメータ」の項目に登録する(S42)と共に、サーバコマンドシートの「状態」を「処理完了」に変更し、処理が完了したことを示して(S43)、もとの図19の処理に戻る。

以上の処理を行うことにより、サーバコマンドを実行し、その結果をコマンド応答としてHTTPサーバ12に送信可能な状態にすることができる。

#### 【0077】

また、サーバコマンドの実行に関する処理としては、図19に示した処理とは独立に、図21に示す処理を実行することも考えられる。この処理においても、HTTPクライアント11のCPU31は、サーバコマンド実行結果生成手段44として機能する。

この場合、CPU31は適当なタイミングで図21のフローチャートに示す処理を開始する。

#### 【0078】

そして、この処理においては、まずサーバコマンドプールに「状態」が「処理待ち」であるサーバコマンドシートがあるか否か判断し(SX1)、なければこのようなサーバコマンドシートが追加されるまで待機する。そして、このようなサーバコマンドシートを発見した場合、その1つを処理対象とし、そのサーバコマンドシートの「状態」を「処理中」に変更する(SX2)。

その後、図20の場合と同様なステップS41乃至S43の処理を行って処理対象のサーバコマンドシートに記載されたサーバコマンドを実行し、これが完了するとステップSX1に戻って処理を繰り返す。

#### 【0079】

以上の処理は、複数のスレッド(例えば4スレッド)で同時に行うようにしてもよい。処理対象となったサーバコマンドシートの「状態」は、「処理待ち」ではないため、複数のスレッドで同時に処理を行っても、1つのサーバコマンドシートを重複して処理対象とってしまうことはない。

#### 【0080】

以上のような処理を行うようにすれば、任意のタイミングでサーバコマンドを実行することができるので、実行に時間のかかるコマンドがあった場合でも、以後の処理が滞ることがない。そして、実行の終了したものから順に、その結果をコマンド応答としてHTTPサーバ12に送信可能な状態にすることができる。

以上で、HTTPクライアント11において実行するこの発明の通信方法に関連する処理の説明を終了する。

#### 【0081】

なお、ここでは図示及び説明を省略したが、HTTPサーバ12についても、この発明の通信方法に関連する部分においては、HTTPクライアント11と同様なハードウェア構成及び機能構成を有し、同様な処理を行う。ただし、サーバとクライアントという役割の差があることから、以下の点でHTTPクライアント11と異なることになる。

#### 【0082】

まず、HTTPリクエスト送信手段46とHTTPレスポンス受信手段47に代えて、HTTPレスポンス送信手段とHTTPリクエスト受信手段を有する。また、クライアントコマンドを受信してこれに対する応答を送信し、サーバコマンドを送信してこれに対する応答を受信することになる。そして、送信メッセージ収集手段45がコマンドやコマンド応答等の読み出しを試みるタイミングは、HTTPリクエストを受信し、その後HTTPリクエストの解析が完了した時点である。HTTPサーバ12側では、HTTPリクエストに対する通信応答として、コマンドやコマンド応答をHTTPクライアント11に送信するためである。

このような処理をHTTPサーバ12側でも行うことにより、この発明の通信方法が実現できる。

#### 【0083】

そして、以上説明してきたこの発明の通信方法によれば、送信元から通信相手に送信すべき動作要求と、通信相手から受信した動作要求に対する動作応答とを、一括して通信相手に送信することができるので、動作要求の送信と動作応答の送信とについて別々にネゴシエーションを行って通信のコネクションを確立する必要がなく、通信のオーバーヘッドを低減して通信効率を高めることができる。

#### 【0084】

また、動作要求と動作応答とをそれぞれ直列化したデータに変換し、構造化言語形式で

記載した送信メッセージに変換しているので、フォーマットの異なる動作要求と動作応答とを容易に結合し、論理的に1つの送信内容として送信することができる。また、受信側でも、受信した内容を容易に個々のメッセージに分離し、それが動作要求であるか動作応答であるかに応じて適切な処理を行うことができる。

#### 【0085】

さらに、通信要求を常に一方の装置から発して通信を行い、その通信相手から通信要求元への動作要求等の送信は、その通信要求に対する応答として行うようにすれば、通信要求を発する側の装置がファイアウォールの内側に設けられているような通信システムであっても、ファイアウォールの存在を意識せずに動作要求及び動作応答の送受信を行うことができる。また、通信要求と通信応答とが対応しているため、通信のレベルでのタイミング管理が容易である。

また、この場合において、上記の一方の装置から通信相手に定期的に通信要求を行うようにすれば、ファイアウォールの外側から内側に向けての情報の送信が長時間に亘って停滞してしまう事態を防止できる。

#### 【0086】

〔SMTPを採用する場合の実施例：図22乃至図27〕

次に通信プロトコルにSMTPを採用する場合の実施例について説明する。この実施例は、上述のHTTPを採用する場合の実施例と共通点が多いので、共通点については説明を省略するか簡単にし、相違点を中心に説明する。

まず、図22に、SMTPを採用する場合の実施例を適用する通信システムの構成例を示す。

#### 【0087】

この通信システムは、図22に示すように、通信装置AとメールサーバA'とを接続したLAN\_Aと、通信装置BとメールサーバB'とを接続したLAN\_Bとを、それぞれファイアウォールAとファイアウォールBとを介してインターネット13に接続して構成している。また、ファイアウォールを介して外部からアクセス可能な位置に、LAN\_A側ではメールサーバAを、LAN\_B側ではメールサーバBをそれぞれ設けている。そして、通信装置Aが第1の通信装置に、通信装置Bが第2の通信装置に該当する。

#### 【0088】

SMTPを用いて通信を行う場合には、通信装置Aと通信装置Bとの間での情報転送は電子メールによって行う。そして、例えば通信装置Bから通信装置Aに情報を送信する場合、図22に破線で示したように、通信装置Bは通信装置Aを宛先とする電子メールを送信し、これをまずメールサーバB'に送信する。すると、各メールサーバB', B, Aを順次介して、通信装置Aが直接アクセスするメールサーバA'まで電子メールが転送される。また、図示は省略したが、通常はメールサーバBとメールサーバAとの間に更に別のメールサーバを介して転送を行うことになる。

そして、一点鎖線で示したように通信装置Aが定期的にメールサーバA'にアクセスすることにより、通信装置A宛の電子メールを受け取ることができ、以上で通信装置Bから通信装置Aへの情報転送が完了する。通信装置Aから通信装置Bへの情報転送は、逆の手順で行うことができ、情報転送に関しては、通信装置Aと通信装置Bとは対称である。ただし、メールサーバA'及びB'を設けることは必須ではなく、通信装置AがメールサーバAと、通信装置BがメールサーバBと直接通信を行うようにしてもよい。また、メール受信用のメールサーバとメール送信用のメールサーバが異なってもよい。

#### 【0089】

このような通信システムにおいて、各LANには外部からアクセス可能な位置にメールサーバを設けていることから、ファイアウォールを越えて電子メールを転送することができる。

なお、この実施例においては、通信装置Aと通信装置Bとが直接ネゴシエーションをした上で通信を行っているわけではないが、相互に情報転送を行うことが可能である。

#### 【0090】

また、通信装置 A 及び通信装置 B も、図 1 に示した第 1 の通信装置及び第 2 の通信装置の場合と同様に、互いの制御管理を行うためのアプリケーションプログラムを実装している。そして、RPC により、互いの実装するアプリケーションプログラムのメソッドに対する処理の依頼である「動作要求」を送信し、この依頼された処理の結果である「動作応答」を取得することができるようになっている。

#### 【0091】

図 23 に、これらの動作要求と動作応答の関係を示すが、動作要求のレベルでは、通信装置 A と通信装置 B との関係は、既に説明した HTTP クライアント 11 と HTTP サーバ 12 との関係と同様に、対称なものである。しかし、SMTP の場合には、通信のレベルでも通信装置 A と通信装置 B とが対称な関係にあることが、HTTP の場合と異なる。

#### 【0092】

図 24 に、この通信システムにおける通信シーケンスの例を示す。この通信シーケンスも、この発明の通信方法の実施形態を示すものである。

上述したように、この通信システムにおいては、通信装置 A と通信装置 B との間の通信は、電子メールを用いて行う。そして、電子メールには、送信元と宛先は存在し、その宛先から送信元に返信を行うことも可能である。しかし、初めの電子メールと返信とは全く独立したものであり、HTTP の場合のような、通信要求と通信応答のような関係はない。

従って、どちらの通信装置から先に通信を行ってもよいし、交互に電子メールを送信しなければならないということもないのであるが、ここでは、通信装置 A から通信装置 B にまず電子メールを送信し、交互に計 3 通の電子メールを送受信する場合の例を示している。

#### 【0093】

これらの電子メールには、送信先（宛先）に対する動作要求（コマンド）及び、送信先から受信したコマンドに対する動作応答（コマンド応答、あるいは単に「応答」）を記載して送信するようにしている。これは、通信装置 A と通信装置 B のどちらが送信元であっても同様である。

従って、例えば通信装置 A 側コマンド a は、電子メール x に記載して転送し、通信装置 B からのコマンド応答をその後の電子メール y に記載して転送することができる。また、通信装置 B 側コマンド c は、電子メール y に記載して転送し、通信装置 A からのコマンド応答をその後の電子メール z に記載して転送することができる。

#### 【0094】

なお、動作要求及び動作応答に対する応答をそれぞれ任意の数ずつ（0 でもよい）1 通の電子メールに記載することができる。そして、1 通の電子メールに記載した内容は、当然論理的に一括して転送する。そして、このようにすることにより、必要な情報を転送するための電子メールの数を減らし、オーバーヘッドを低減して通信の効率化を図っている。

#### 【0095】

次に、通信装置 A 及び通信装置 B における、コマンドやコマンド応答を結合し、また分離する処理を行うための機能構成及びその処理の手順について説明する。ハードウェア構成については、HTTP を用いる実施例の説明において図 7 に示した HTTP クライアント 11 の場合と同様なものを使用することができる。

#### 【0096】

図 25 は、通信装置 A の機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す図 8 と対応する機能ブロック図である。

図 25 に示す機能のうち、通信装置 A 側コマンドプール 51、通信装置 B 側コマンドプール 52、通信装置 A 側コマンド生成手段 53、通信装置 B 側コマンド実行結果生成手段 54 は、それぞれ図 8 に示したクライアントコマンドプール 41、サーバコマンドプール 42、クライアントコマンド生成手段 43、サーバコマンド実行結果生成手段 44 と対応する機能であり、装置の名称が異なることにより名称を変更したものである。

**【0097】**

また、送信メッセージ収集手段45及び受信メッセージ分配手段48は、図8に示した同名の手段と対応するものである。ただし、送受信するためのデータの形式が、SMTPに対応した形式である点が図8に示した例とは異なる。しかし、コマンドやコマンド応答に対応する、個々のSOAPメッセージについては、図8の場合と同様なものである。

メール送信手段56及びメール受信手段57については、送受信に使用するプロトコルが異なることに伴って、図8に示したHTTPリクエスト送信手段46及びHTTPレスポンス受信手段47とは異なるものである。

**【0098】**

すなわち、メール送信手段56は、送信メッセージ収集手段45が生成した送信メッセージを含む、通信装置B宛の電子メールを生成し、メールサーバA'に送信する機能を有する。なお、それ以降の電子メールの転送には、通信装置Aは関与しない。また、1つの電子メールに送信メッセージをいくつ含めてもよいし、コマンド応答に係る送信メッセージと通信装置A側コマンドに係る送信メッセージとを任意に混在させることもできる点等は、図8に示した例の場合と同様である。

**【0099】**

メール受信手段57は、定期的にメールサーバA'にアクセスして通信装置A宛での新着メールの有無を調べ、新着メールがあった場合にこれを受信する機能を有する。そしてここでは、受信する電子メールには、通信装置B側コマンド及びそのコマンドと関連付けられたコマンドIDを含む受信メッセージと、通信装置A側コマンドに対する応答及びそのコマンドと関連付けられたコマンドIDを含む受信メッセージとが、任意に混在して含まれている。

**【0100】**

次に、このような機能を有する通信装置Aが通信装置B宛てに送信する電子メールの例を図26に示す。

この電子メールは、図11に示したHTTPリクエストの場合と同様に、ボディ部としてMIME (Multipurpose Internet Mail Extension) に従ったマルチパートのメッセージが記載され、この各パートには、それぞれSOAPエンベロープが埋め込まれている。すなわち、ボディの部分はHTTPリクエストの場合と同様なものになっている。

**【0101】**

しかし、ヘッダ部分はHTTPリクエストの場合とは異なり、電子メールの送信元アドレスを示すFrom、宛先アドレスを示すTo、表題を示すSubject等の情報が記載されている。

通信装置Bから通信装置Aに送信する電子メールについては、ヘッダのうちFromの内容とToの内容を入れ替えたものになる。

また、これらの電子メールに記載されるSOAPエンベロープの内容は、HTTPリクエストやHTTPレスポンスの場合と同様なものである。ただし、各パートのエンティティヘッダの部分は、データのエンコード方式が異なるためHTTPの場合とは一部が異なる。

**【0102】**

次に、以上説明したような構成及び機能を有する通信装置Aにおいて実行する処理について、図27及び図28のフローチャートを用いて説明する。これらのフローチャートに示す処理は、通信装置AのCPUが所要の制御プログラムを実行することによって行うものである。

**【0103】**

まず、図27にメッセージ送信時の処理の基本動作のフローチャートを示す。

通信装置AのCPUは、送信メッセージ収集手段45が通信装置A側コマンドやコマンド応答等の読み出しを試みるタイミングになると、図27のフローチャートに示す処理を開始する。

そして、まず通信装置A側コマンドの収集処理を行う(S51)。この処理は、通信装

置 A 側コマンドプール 51 から通信装置 B に送信すべき通信装置 A 側コマンドを収集する処理であり、収集したデータから SOAP エンベロープのパートを生成する処理を含む。

#### 【0104】

次に、通信装置 B 側コマンドに対する応答である通信装置 B 側コマンド実行結果の収集処理を行う (S52)。この処理は、通信装置 B 側コマンドプール 52 から通信装置 B に送信すべきコマンド応答を収集する処理であり、やはり収集したデータから SOAP エンベロープのパートを生成する処理を含む。

その後、ステップ S51 及び S52 の処理で生成したパートを 1 つにマージして、すべてのパートを含む電子メールを生成し (S53)、その電子メールを通信装置 B に宛てて送信して (S54)、処理を終了する。

#### 【0105】

以上の処理において、ステップ S51 及び S52 では CPU31 は送信メッセージ収集手段 45 として機能し、ステップ S53 及び S54 ではメール送信手段 56 として機能する。

これらの処理は、図 17 に示したステップ S11 乃至 S14 の処理と対応するものであるが、SMTP の場合には、電子メールの送信と受信の間には、通信要求と通信応答のような関係はないため、処理は一旦ここで終了し、電子メールを受信する場合には、別途図 28 に示す処理を行う。

#### 【0106】

図 28 には、メッセージ受信時の処理の基本動作のフローチャートを示す。

通信装置 A の CPU は、定期的にメールサーバ A' にアクセスし、通信装置 A 宛ての新着の電子メールがあると、図 28 のフローチャートに示す処理を開始する。

この処理においては、まず新着の電子メールを受信する (S61)。そして、受信した電子メールのボディ (本文) を、各パートに分割する (S62)。ここで、各パートへの分割は、「MIME\_boundary」で区分された要素に分割することであり、またここで全てのパートに関して分解する。

#### 【0107】

そしてその後、全てのパートを順に対象としてステップ S63 乃至 S65 の処理を繰り返す。この処理においては、まず対象のパートが通信装置 B 側コマンドを記載したパートか否か判断する (S63)。そして、通信装置 B 側コマンドであれば通信装置 B 側コマンド登録処理を行う (S64)。また、通信装置 B 側コマンドでないときは、通信装置 A 側コマンドに対する応答が記載されたパートであるので、応答通知処理を行う (S65)。

#### 【0108】

ステップ S64 又は S65 の後は、ステップ S63 に戻り、次のパートを対象として処理を繰り返す。そして、全てのパートについてこれらの処理を行った時点で、図 28 のフローチャートに示す処理を終了する。

以上の処理においては、ステップ S61 では CPU31 はメール受信手段 57 として機能し、ステップ S62 乃至 S65 では受信メッセージ分配手段 48 として機能する。

これらの処理は、図 17 に示したステップ S15 乃至 S19 の処理と対応するものである。

#### 【0109】

また、通信装置 B 側コマンドの実行に関する処理については、図 20 及び図 21 を用いて説明したサーバコマンドの実行に関する処理と同様なものである。

以上で、通信装置 A において実行する、この発明の通信方法に関連する処理の説明を終了する。

なお、通信装置 B の機能及び通信装置 B において実行する処理については、通信装置 A 側要求と通信装置 B 側要求との意味合いが反対になる点以外は、通信装置 A の場合と全く同じであるので、説明を省略する。

#### 【0110】

以上説明してきたように、この発明の通信方法は、SMTP のように、情報の送信と返

信とが必ずしも対応しないプロトコルを使用して通信を行う場合にも適用できる。そして、この場合にも、送信元から通信相手に送信すべき動作要求と、通信相手から受信した動作要求に対する動作応答とを、一括して通信相手に送信することができるので、動作要求の送信と動作応答の送信とについて別々の単位で転送を行う必要がなく、通信のオーバーヘッドを低減して通信効率を高めることができる。

また、通信に電子メールを用いれば、ファイアウォールの内側へも容易に情報を転送することができる。

#### 【0111】

〔各実施例の変形例：図29〕

次に、上述した各実施例の変形例について説明する。

まず、上述した各実施例では、説明を簡単にするために2つの通信装置からなる通信システムにこの発明を適用する例について説明したが、この発明は、さらに多くの通信装置からなる通信システムに適用することも当然可能である。

例えば、HTTPを採用する場合の実施例は、図29に示すような通信システムにも適用することができる。

#### 【0112】

この通信システムは、通信機能を備えた通信装置である管理装置102と、同じく通信機能を備えた通信装置である被管理装置100とをインターネット13を含むネットワークを介して接続し、管理装置102によって被管理装置100を管理する遠隔管理システムとして構成している。

ここで、被管理装置100の具体例としては、プリンタ、スキャナ、複写機、あるいはこれらの機能を兼ね備えたデジタル複合機等の画像処理装置を始めとし、通信機能を備えたネットワーク家電、自動販売機、医療機器、電源装置、空調システム、ガス・水道・電気等の計量システム等や、ネットワークに接続可能なコンピュータ等も含め、通信機能を備える各種電子装置が考えられる。

#### 【0113】

また、この遠隔管理システムにおいては、管理装置102と被管理装置100との間の通信を仲介する通信装置である仲介装置101を設けており、管理装置102と被管理装置100とはこの仲介装置101を介して通信を行う。

そして、仲介装置101及び被管理装置100は、その利用環境に応じて多様な階層構造を成す。例えば、図29に示す設置環境Aでは、管理装置102とHTTPによるコネクションを確立できる仲介装置101aが、被管理装置100a及び100bを従える単純な階層構造になっているが、設置環境Bでは、4台の被管理装置100を設置するため、1台の仲介装置101を設置しただけでは負荷が大きくなる。

#### 【0114】

そのため、管理装置102とHTTPによるコネクションを確立できる仲介装置101bが、被管理装置100c及び100dだけでなく、他の仲介装置101cを従え、この仲介装置101cが被管理装置100e及び100fを更に従えるという階層構造を形成している。この場合、被管理装置100e及び100fを遠隔管理するために管理装置102から発せられた情報は、仲介装置101bとその下位のノードである仲介装置101cとを経由して、被管理装置100e又は100fに到達することになる。なお、各設置環境には、セキュリティ面を考慮し、ファイアウォール14を設置している。

#### 【0115】

このような遠隔管理システムにおいても、仲介装置101をHTTPクライアント、管理装置102をHTTPサーバとして取り扱うことにより、この発明の通信方法を適用することができる。

なお、この場合において、管理装置102から被管理装置100にコマンドを送信しようとする場合、仲介装置101に対して、「被管理装置100にコマンドを送信せよ」というコマンドを送信し、コマンドに応じた動作として被管理装置100に対してコマンドを送信させることが考えられる。また、宛先として被管理装置100のいずれかを指定し

て仲介装置 101 に対してコマンドを送信し、仲介装置 101 に、自己以外が宛先となっているコマンドを、その宛先に対して転送させることも考えられる。

#### 【0116】

また、3つ以上のノード間でコマンドやコマンド応答の送受信を行う場合、コマンドの発信元と宛先とを把握できるように、これらの情報もコマンドやコマンド応答のメッセージに含め、またコマンドシートにも記載して管理するようにするとよい。

また、メールサーバを設ければ、SMTPを採用する場合の実施例も、このような通信システムに適用することができる。

#### 【0117】

さらに、この発明には、上記以外の変形を適用することも可能である。例えば、クライアントコマンドプール 41 及びサーバコマンドプール 42 に登録するクライアントコマンドシート及びサーバコマンドシートを、XMLドキュメントとして記載するようにしてもよい。「入力パラメータ」をデシリアライズする前のXMLドキュメントとして保存したり、「出力パラメータ」をシリアライズした後のXMLドキュメントとして保存したりしてもよい。

また、HTTPサーバ 12 へ送信するコマンドやコマンド応答の情報量に制限を設けても構わない。HTTPサーバ 12 から受信するコマンドやコマンド応答の情報量に制限を設けても構わない。特に、受信するコマンドの情報量を制限するようにすると、受信側がメモリ容量の限られた装置である場合にメモリの使用量を抑えることができる。

#### 【0118】

また、上述した実施例においては、RPCを実現する上位プロトコルとしてSOAPを採用し、アプリケーションは直接プールを操作してRPCを実現しているが、アプリケーションとプールとの間にCORBA (Common Object Request Broker Architecture) や J A V A (登録商標) R M I (Remote Method Invocation) とのブリッジ (メッセージ変換機能) を備えることによってアプリケーションの開発効率をさらに向上させてもよい。

すなわち、上述した実施例における、HTTPクライアント 11 と HTTPサーバ 12 との間等でのコマンド及びこれに対するコマンド応答のやり取りは、XMLで記述されたSOAPメッセージにより行うこととしているが、これに限るものでなく、他の形式で記述されていてもよい。

#### 【0119】

また、上述した実施例において、SOAP標準のプロトコルだけでなく、SOAPとMIMEマルチパートを組み合わせた独自のプロトコルをもこれに加えて採用することにより、HTTPリクエスト、或いはHTTPレスポンスに含まれるSOAPエンベロープを全く独立したものとして扱うこととするが、SOAPの関連仕様として定義されたSOAPアタッチメントによって、HTTPレスポンスに含まれる第1パートのSOAPエンベロープに、第2パート以降のSOAPエンベロープへのリンクを埋め込んでこれらを関連付けて引き渡す構成にしてもよい。SMTPを用いる電子メールの場合にも同様である。

#### 【0120】

更に、SOAP等の上位プロトコルの下位に位置するデータ通信のプロトコルとして、ここではHTTPあるいはSMTPを採用した例について説明したが、この下位プロトコルについても、FTP (File Transfer Protocol) 等の他のプロトコルを採用してもよい。

さらにまた、この発明を適用する通信システムについても、以上説明したものに限られることはない。

#### 【産業上の利用可能性】

#### 【0121】

以上説明してきたように、この発明の通信方法によれば、複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する場合において、通信の効率を上げることができる。従って、この発明を、複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する通信システムに適用することにより、通信の負荷



が小さい通信システムを構成することができる。

【図面の簡単な説明】

【0122】

【図1】この発明の通信方法を適用する通信システムの構成例を示す図である。

【図2】図1に示した通信システムにおける動作要求と動作応答の関係を示す図である。

【図3】通信プロトコルとしてHTTPを採用する場合のこの発明の通信方法の実施例を適用する通信システムの構成例を示す図である。

【図4】図3に示した通信システムにおける動作要求と動作応答の関係を示す図である。

【図5】図3に示した通信システムにおける通信シーケンスの例を示す図である。

【図6】その別の例を示す図である。

【図7】図3に示したHTTPクライアント及びHTTPサーバのハードウェア構成例を示す図である。

【図8】図3に示したHTTPクライアントの機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す機能ブロック図である。

【図9】図8に示したクライアントコマンドプールに記憶させるクライアントコマンドシートにおけるデータ構造の例を示す図である。

【図10】図8に示したサーバコマンドプールに記憶させるサーバコマンドシートにおけるデータ構造の例を示す図である。

【0123】

【図11】図3に示したHTTPクライアントがHTTPサーバに送信するHTTPリクエストの例を示す図である。

【図12】図3に示したHTTPクライアントがHTTPサーバから受信するHTTPレスポンスの例を示す図である。

【図13】クライアントコマンドを記載したパートの例を示す図である。

【図14】クライアントコマンドに対する応答を記載したパートの例を示す図である。

【図15】サーバコマンドを記載したパートの例を示す図である。

【図16】サーバコマンドに対する応答を記載したパートの例を示す図である。

【図17】図3に示したHTTPクライアントにおける、メッセージの収集及び分配処理の基本動作を示すフローチャートである。

【図18】図17のステップS11乃至S14の部分の処理をより詳細に示すフローチャートである。

【図19】図17のステップS15以下の部分の処理をより詳細に示すフローチャートである。

【図20】図3に示したHTTPクライアントにおける、サーバコマンドの実行に関する処理の一例を示すフローチャートである。

【0124】

【図21】その別の例を示すフローチャートである。

【図22】通信プロトコルとしてSMTPを採用する場合のこの発明の通信方法の実施例を適用する通信システムの構成例を示す図である。

【図23】図22に示した通信システムにおける動作要求と動作応答の関係を示す図である。

【図24】図22に示した通信システムにおける通信シーケンスの例を示す図である。

【図25】図22に示した通信装置Aの機能のうち、コマンド及びコマンド応答に関する処理を行うための機能の構成を示す、図8と対応する機能ブロック図である。

【図26】図22に示した通信装置Aが通信装置B宛てに送信する電子メールの例を示す図である。

【図 27】 図 22 に示した通信装置 A における、メッセージ送信時の処理の基本動作を示すフローチャートである。

【図 28】 同じく、メッセージ受信時の処理の基本動作のフローチャートを示すフローチャートである。

【図 29】 この発明の通信方法の実施例を適用する通信システムの別の構成例を示す図である。

【符号の説明】

【0125】

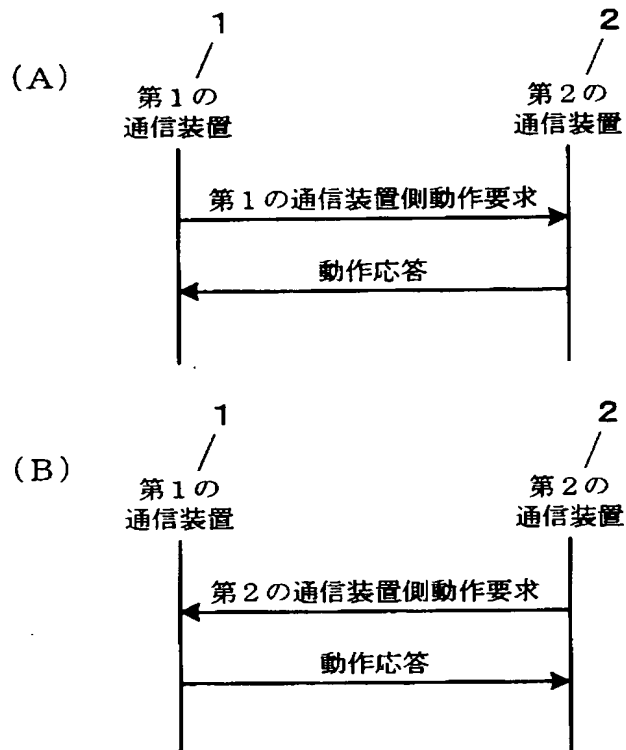
- |                         |                |
|-------------------------|----------------|
| 1：第 1 の通信装置             | 2：第 2 の通信装置    |
| 10：ネットワーク               | 11：HTTP クライアント |
| 12：HTTP サーバ             | 13：インターネット     |
| 14：ファイアウォール             | 31：CPU         |
| 32：ROM                  | 33：RAM         |
| 34：SD カード               | 35：NIC         |
| 41：クライアントコマンドプール        |                |
| 42：サーバコマンドプール           |                |
| 43：クライアントコマンド生成手段       |                |
| 44：サーバコマンド実行結果生成手段      |                |
| 45：送信メッセージ収集手段          |                |
| 46：HTTP リクエスト送信手段       |                |
| 47：HTTP レスポンス受信手段       |                |
| 48：受信メッセージ分配手段          |                |
| 51：通信装置 A 側コマンドプール      |                |
| 52：通信装置 B 側コマンドプール      |                |
| 53：通信装置 A 側コマンド生成手段     |                |
| 54：通信装置 B 側コマンド実行結果生成手段 |                |
| 56：メール送信手段              | 57：メール受信手段     |
| 100：被管理装置               | 101：仲介装置       |
| 102：管理装置                |                |

【書類名】 図面

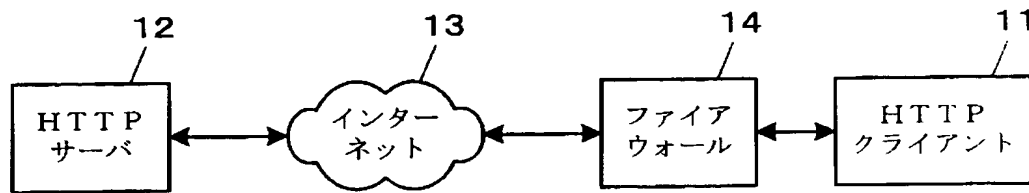
【図 1】



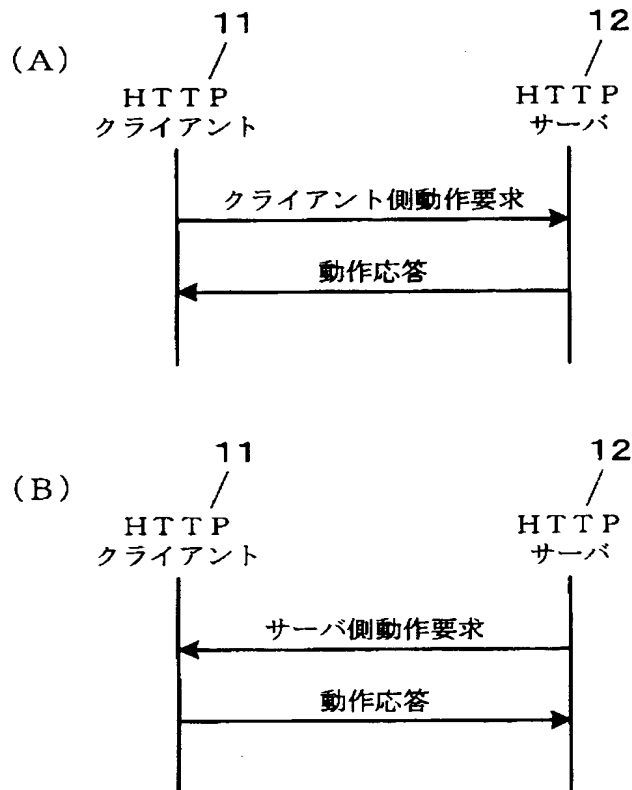
【図 2】



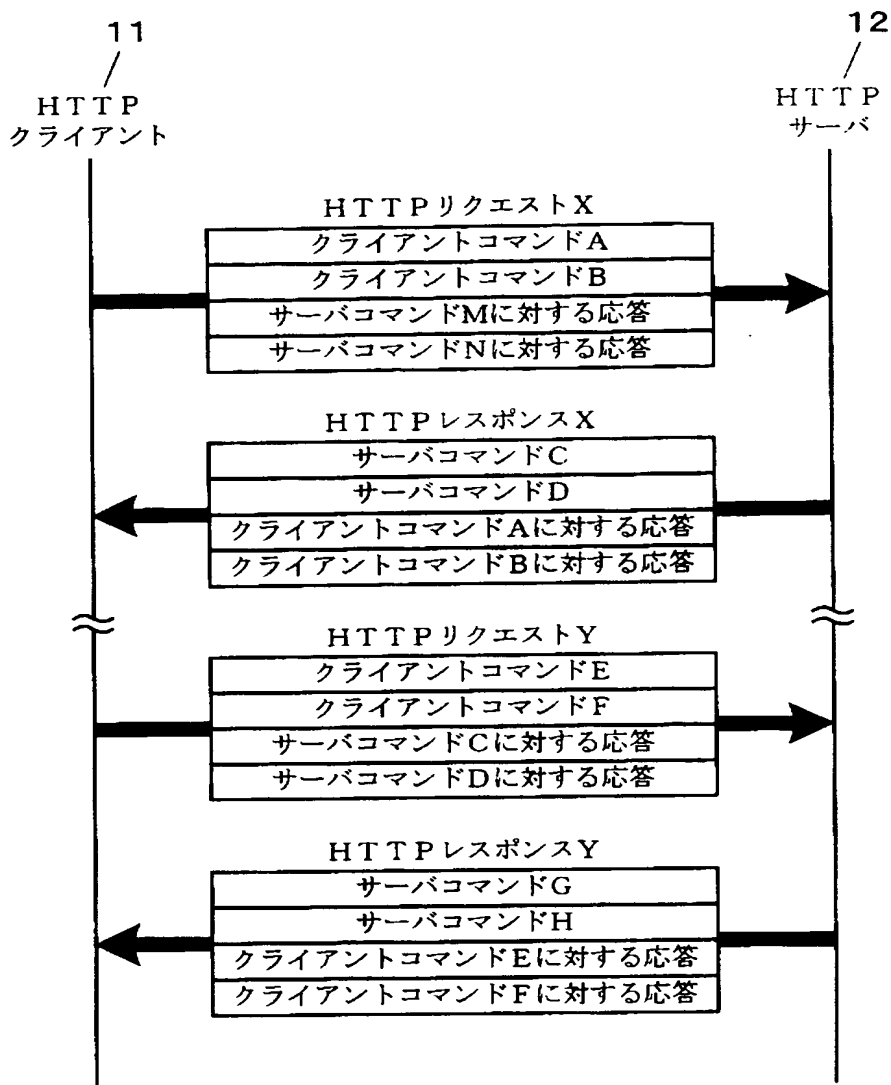
【図 3】



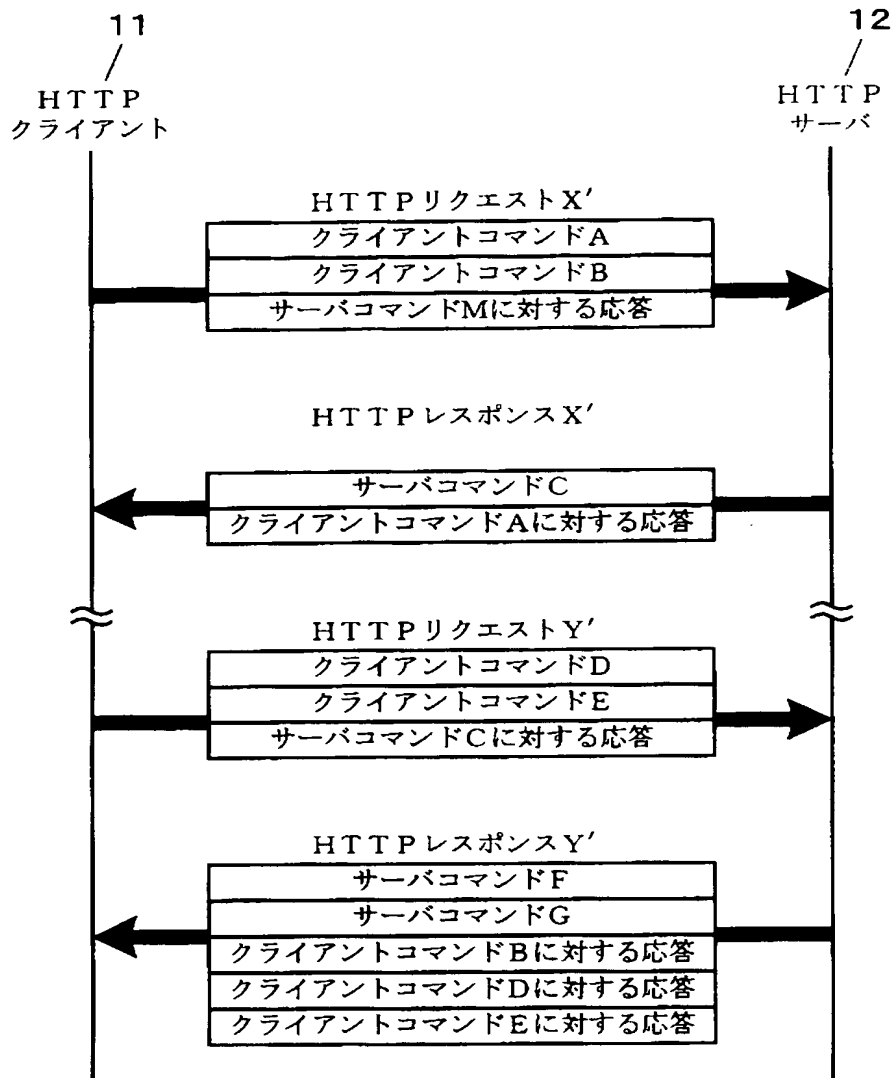
【図 4】



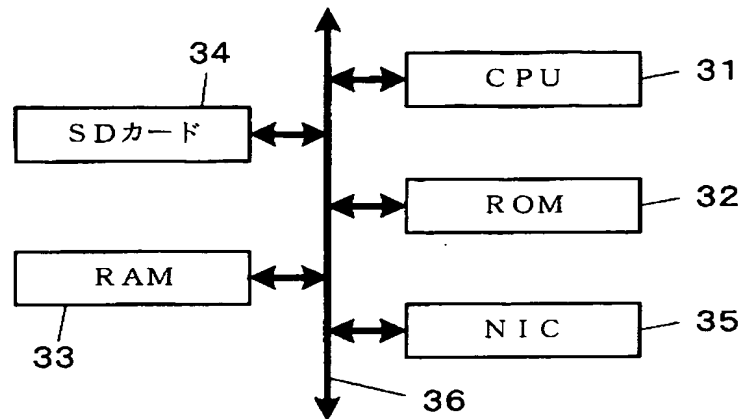
【図 5】



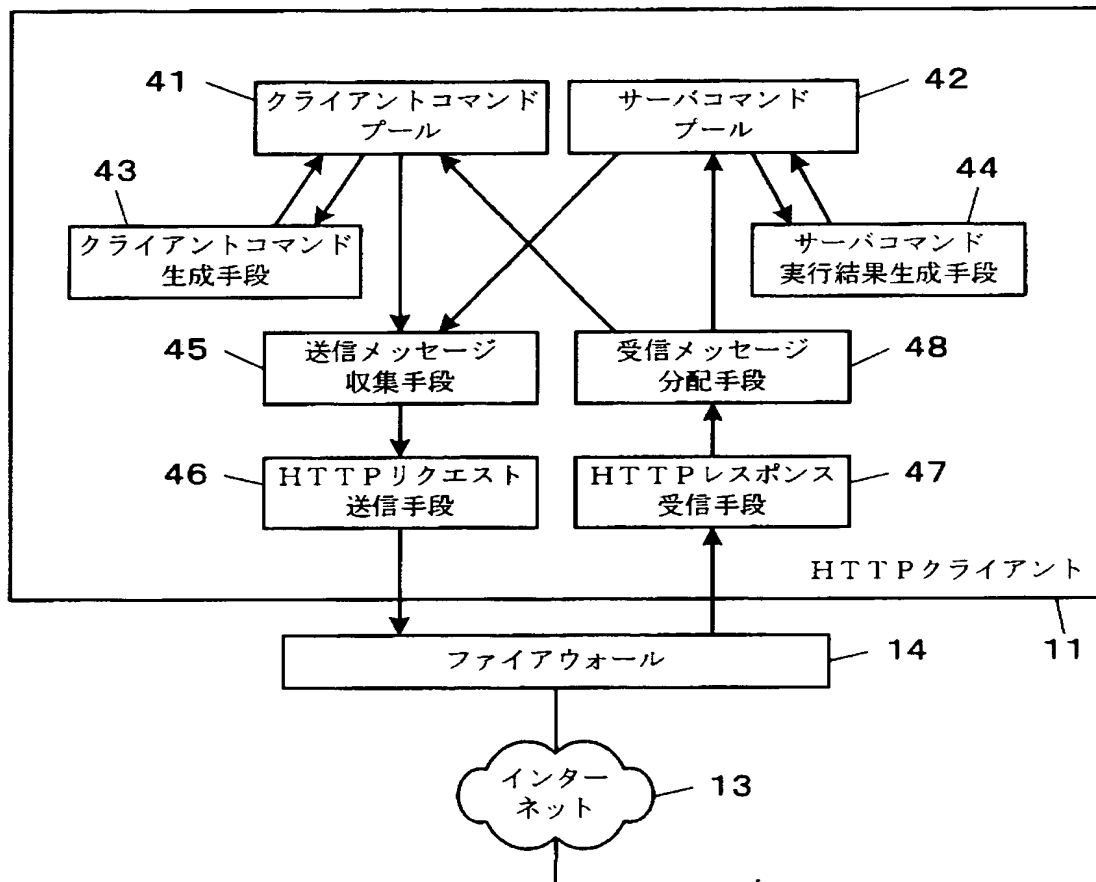
【図6】



【図7】



【図8】



【図 9】

## クライアントコマンドシート

コマンドID  
メソッド名（異常通知等）  
入力パラメータ（異常内容等）  
状態（初期値：未処理）  
クライアントコマンド実行結果の通知先  
出力パラメータ（応答取得まで空）

【図 10】

## サーバコマンドシート

コマンドID  
メソッド名（カウンタ取得等）  
入力パラメータ  
状態（初期値：未処理）  
出力パラメータ（処理終了まで空）  
サーバコマンドの通知先



【図 11】

## HTTP リクエスト

```
POST /aaa HTTP/1.1
Content-Type: multipart/mixed; boundary=MIME_boundary
Content-Length: nnnn
```

```
--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Request
SOAPAction: "SOAP Action URI"
```

第 1 パート

```
<s:Envelope>
  <!--SOAP Request-->
</s:Envelope>
```

```
--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Response
```

第 2 パート

```
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
```

```
--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Response
```

第 3 パート

```
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
```

```
--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Response
```

第 4 パート

```
<s:Envelope>
  <!--SOAP Response-->
</s:Envelope>
```

```
--MIME_boundary--
```

【図 12】

## HTTPレスポンス

```
HTTP/1.1 200 OK
Content-Type:multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Request
SOAPAction:"SOAP Action URI"
<s:Envelope>
  <--SOAP Request-->
</s:Envelope>
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
<s:Envelope>
  <--SOAP Response-->
</s:Envelope>
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
<s:Envelope>
  <--SOAP Response-->
</s:Envelope>
--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response
<s:Envelope>
  <--SOAP Response-->
</s:Envelope>
--MIME_boundary--
```

第1パート

第2パート

第3パート

第4パート

## 【図 13】

クライアントコマンドのパート例

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Request
SOAPAction: "http://www.foo.com/server/errorNotification"

<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:ns="http://www.foo.com/server"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

  <s:Header>
    <n:要求ID>12345</n:要求ID>
  </s:Header>
  </s:Body>
  <ns:異常通知>
    <エラーID>1111</エラーID>
    <説明>ハードディスクドライブ異常</説明>
  </ns:異常通知>
</s:Body>
</s:Envelope>
```

## 【図 14】

クライアントコマンドに対する応答のパート例

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Response

<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:ns="http://www.foo.com/server"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

  <s:Header>
    <n:要求ID>12345</n:要求ID>
  </s:Header>
  </s:Body>
  <ns:異常通知Response>
    <受信結果>OK</受信結果>
  </ns:異常通知Response>
</s:Body>
</s:Envelope>
```

【図 1 5】

サーバコマンドのパート例

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Request
SOAPAction: "http://www.foo.com/client/getTemperature"

<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:nc="http://www.foo.com/client"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

  <s:Header>
    <n:要求 I D>98765</n:要求 I D>
  </s:Header>
  </s:Body>
    <nc:温度センサ値取得>
      <センサ I D>3</センサ I D>
    </nc:温度センサ値取得>
  </s:Body>
</s:Envelope>
```

【図 1 6】

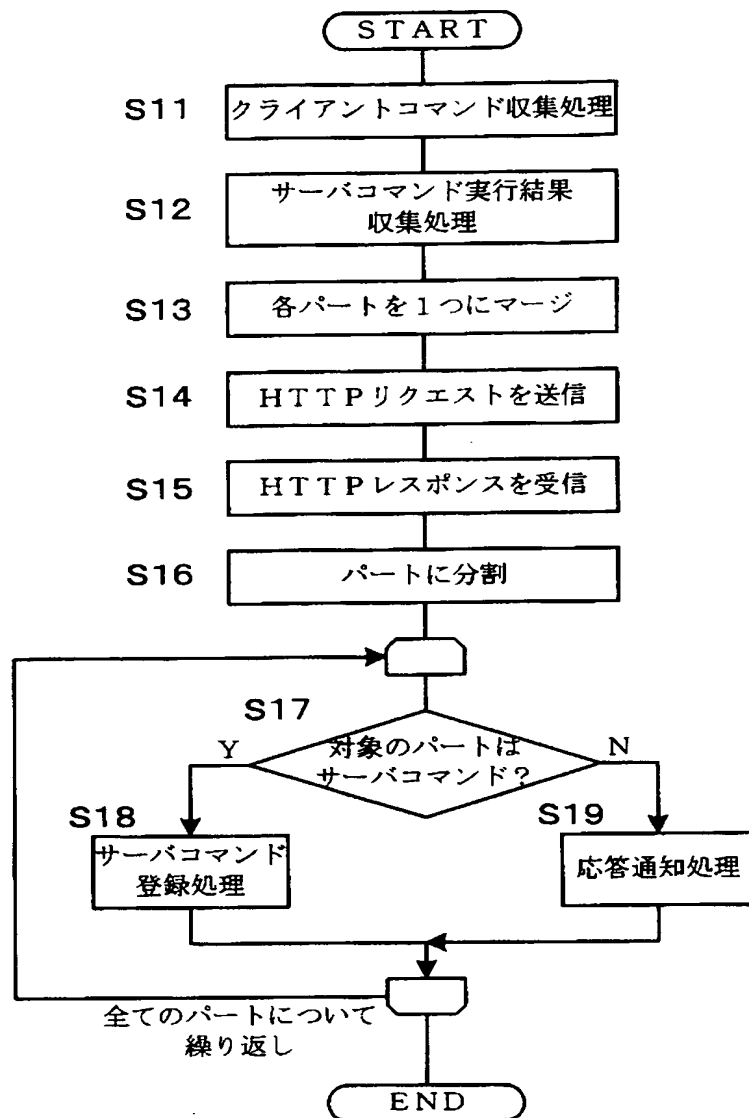
サーバコマンドに対する応答のパート例

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type: Response

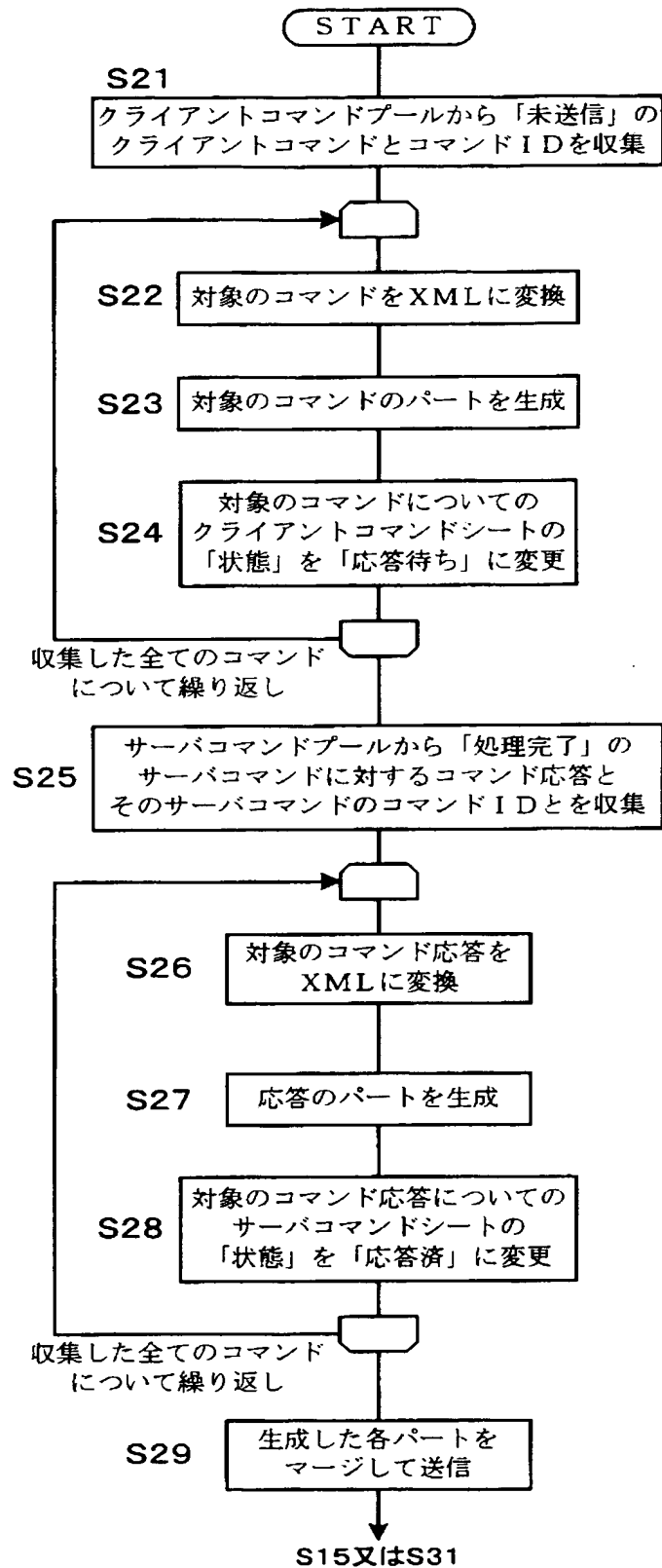
<s:Envelope
  xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:n="http://www.foo.com/header"
  xmlns:nc="http://www.foo.com/client"
  s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">

  <s:Header>
    <n:要求 I D>98765</n:要求 I D>
  </s:Header>
  </s:Body>
    <nc:温度センサ値取得Response>
      <温度>52</温度>
    </nc:温度センサ値取得Response>
  </s:Body>
</s:Envelope>
```

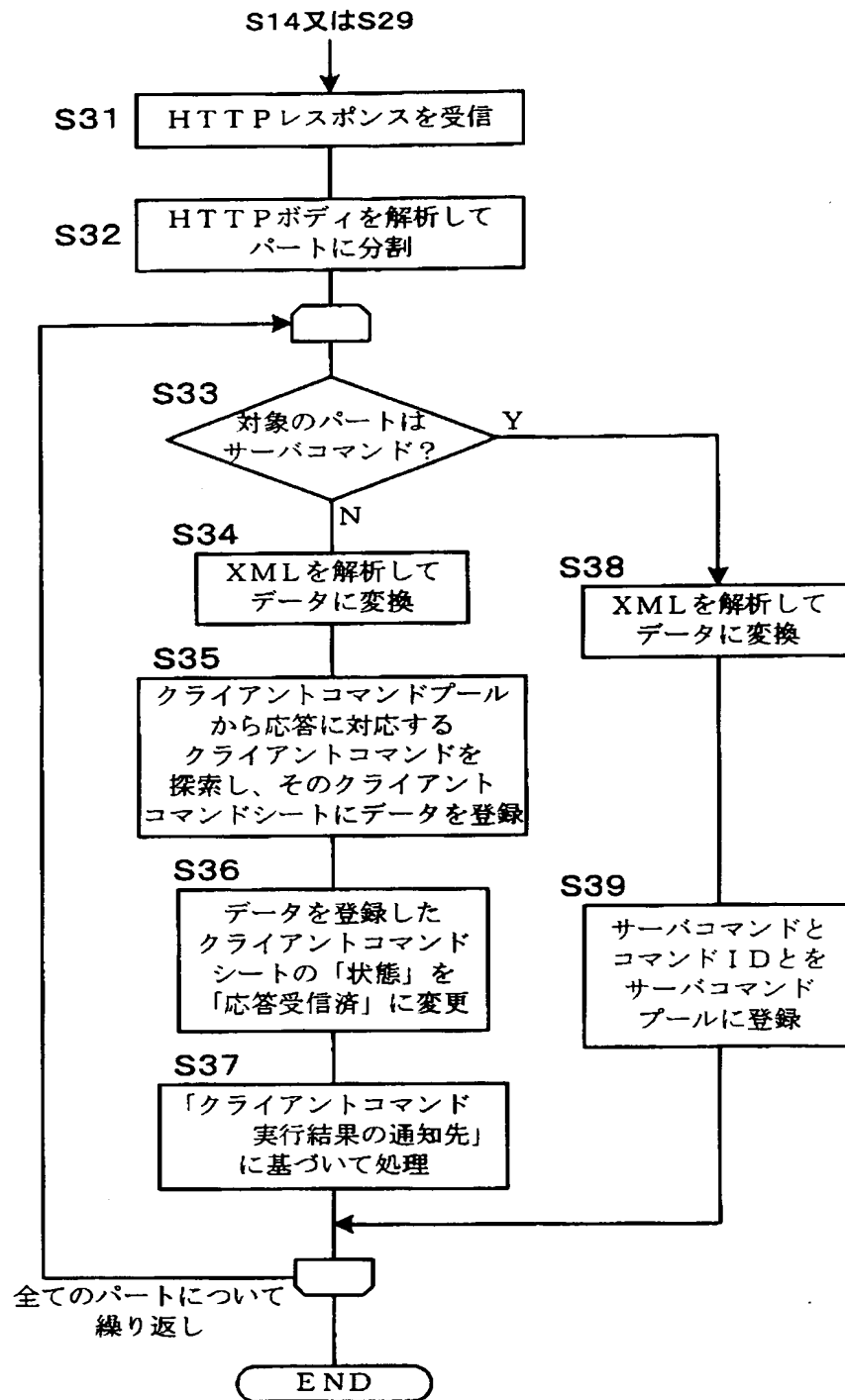
【図 17】



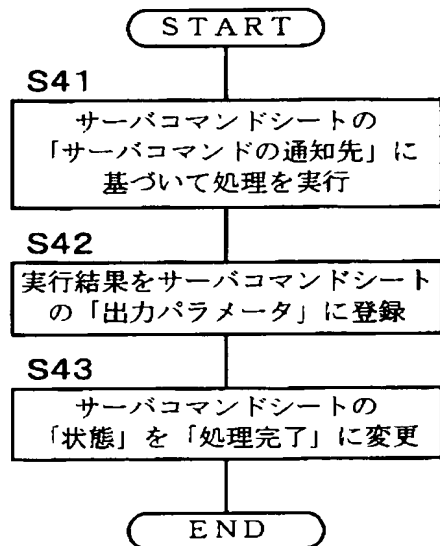
【図18】



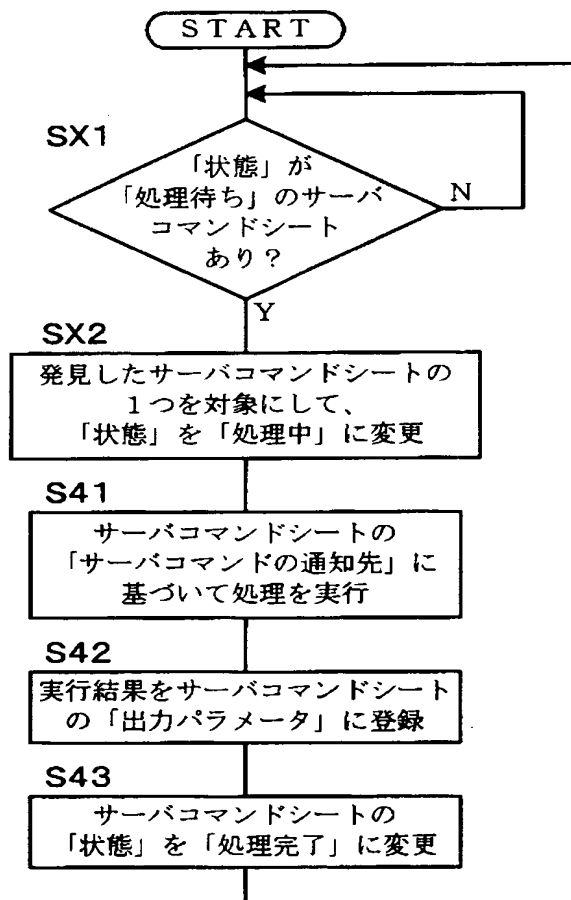
【図 19】



【図 20】

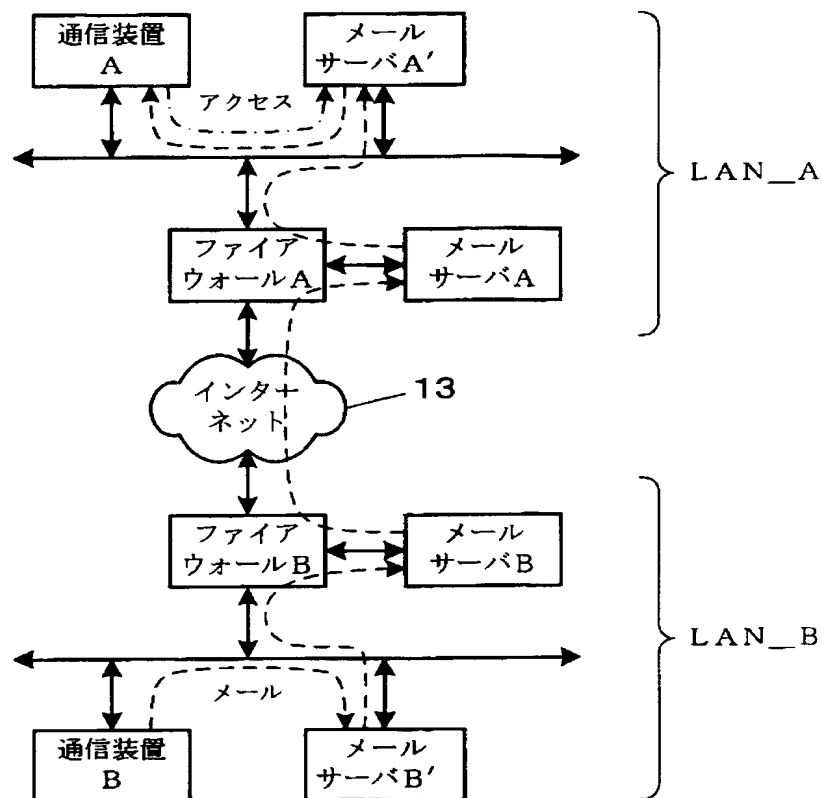


【図 21】



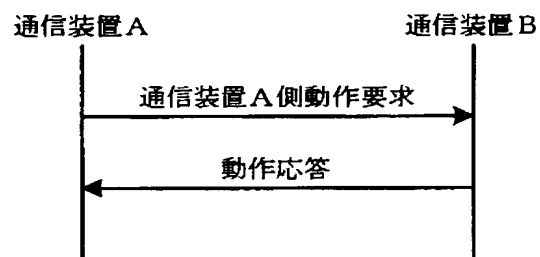


【図 22】

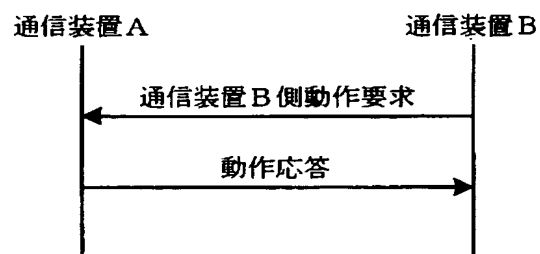


【図 23】

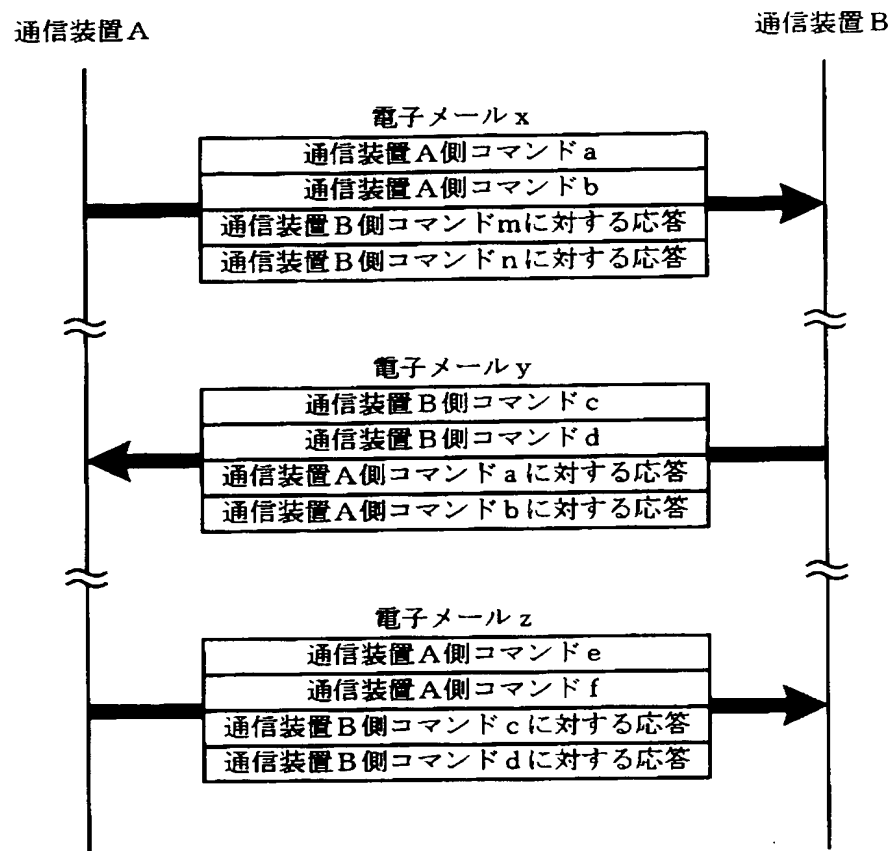
(A)



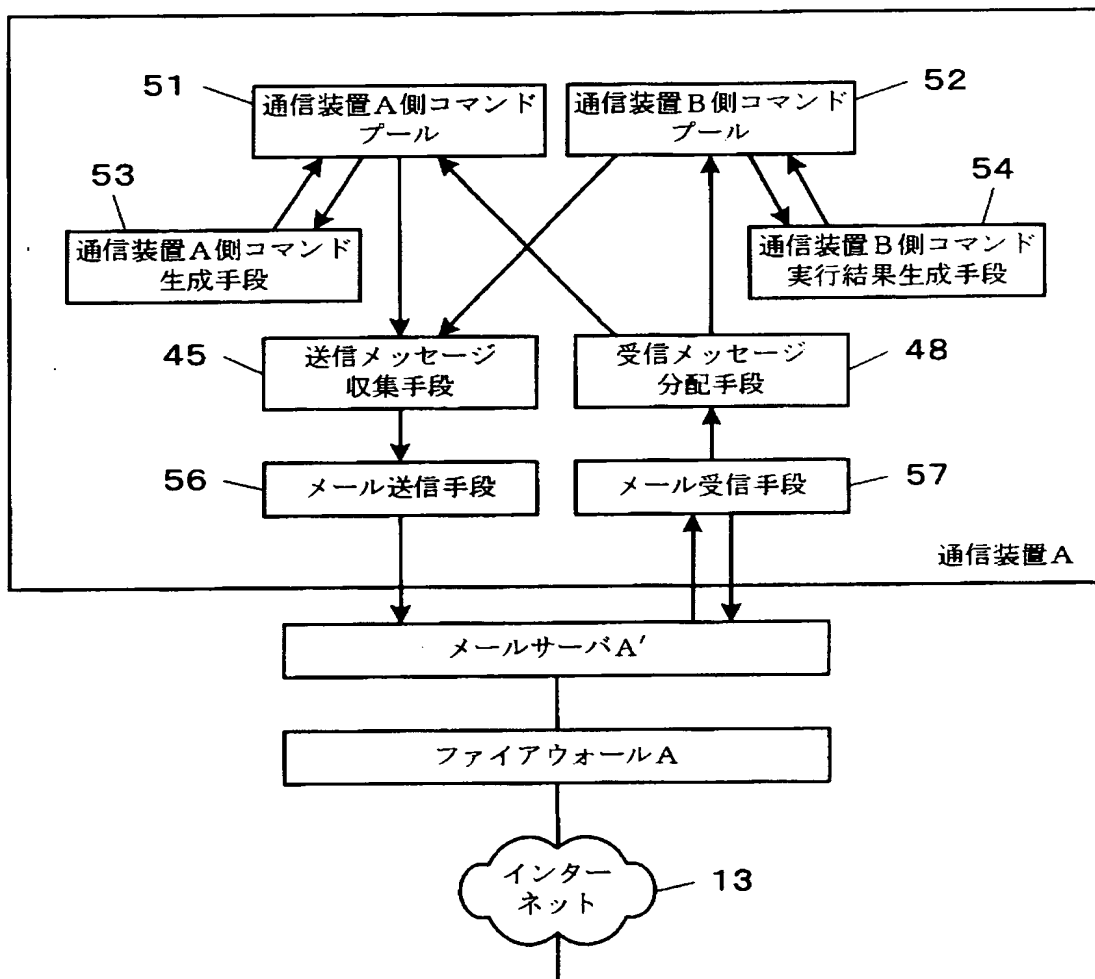
(B)



【図 24】



【図 25】



## 【図 26】

```
From: deviceA@foo.com
To: deviceB@bar.com
Subject: Multi Message #00000001
Date: Wed, 30 Jul 2003 10:00:00 +0900
Content-Type: multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn
```

```
--MIME_boundary
```

```
Content-Type:text/xml;charset=iso-2022-jp
```

```
Content-Transfer-Encoding:7bit
```

```
SOAPAction:"SOAP Action URI"
```

```
X-SOAP-Type:Request
```

```
<s:Envelope>
```

```
<--SOAP Request-->
```

```
</s:Envelope>
```

第1パート

```
--MIME_boundary
```

```
Content-Type:text/xml;charset=iso-2022-jp
```

```
Content-Transfer-Encoding:7bit
```

```
X-SOAP-Type:Response
```

```
<s:Envelope>
```

```
<--SOAP Response-->
```

```
</s:Envelope>
```

第2パート

```
--MIME_boundary
```

```
Content-Type:text/xml;charset=iso-2022-jp
```

```
Content-Transfer-Encoding:7bit
```

```
X-SOAP-Type:Response
```

```
<s:Envelope>
```

```
<--SOAP Response-->
```

```
</s:Envelope>
```

第3パート

```
--MIME_boundary
```

```
Content-Type:text/xml;charset=iso-2022-jp
```

```
Content-Transfer-Encoding:7bit
```

```
X-SOAP-Type:Response
```

```
<s:Envelope>
```

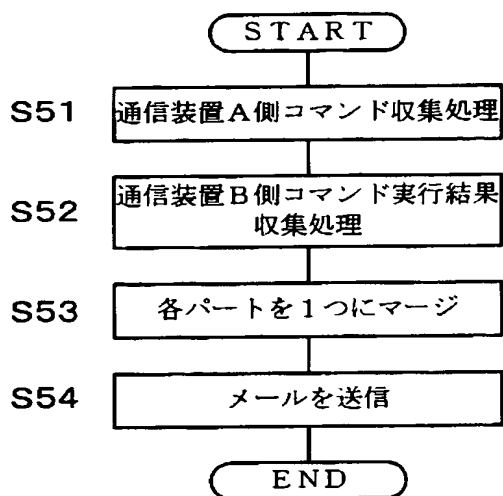
```
<--SOAP Response-->
```

```
</s:Envelope>
```

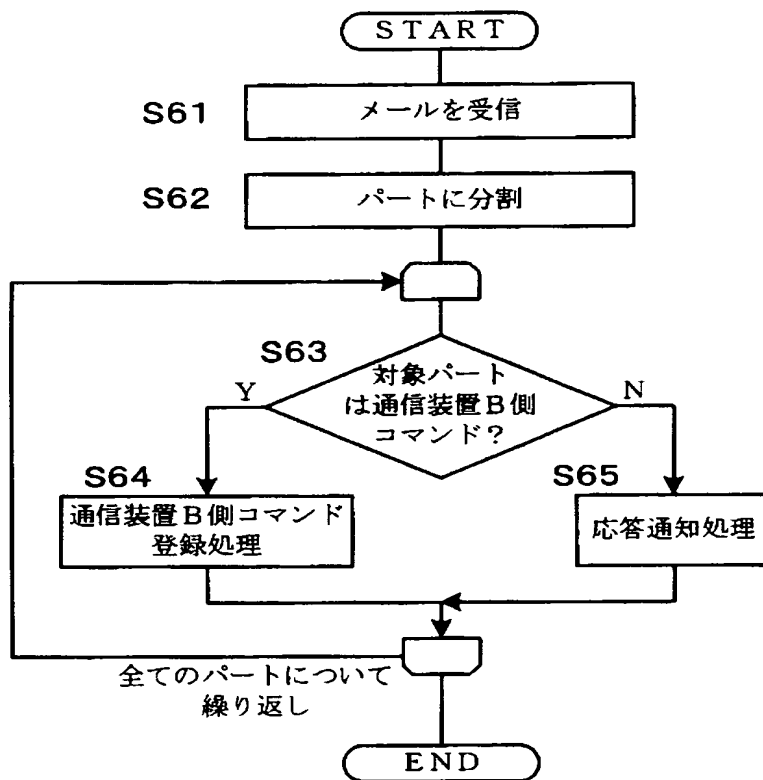
第4パート

```
--MIME_boundary--
```

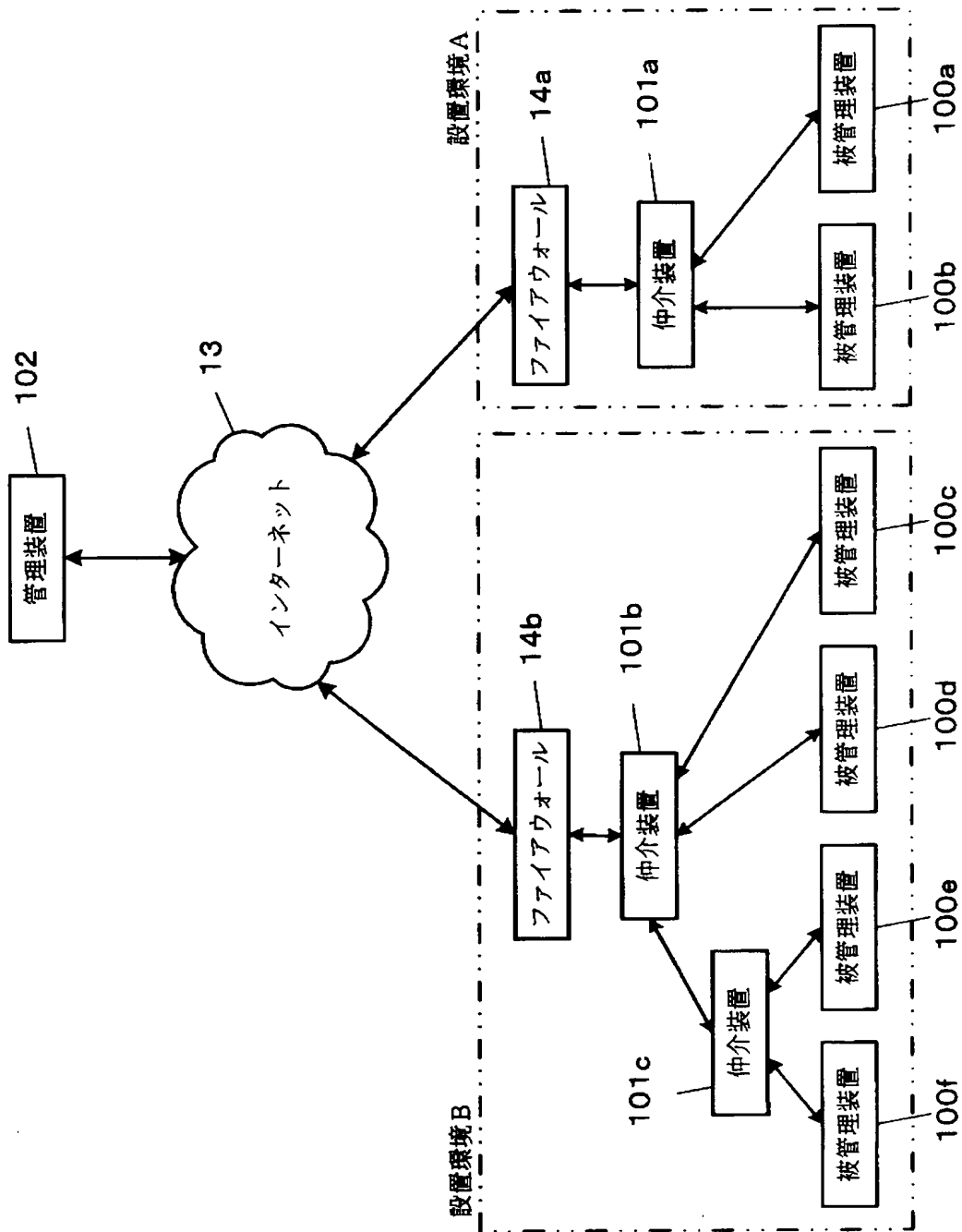
【図 27】



【図 28】



【図 29】



【書類名】 要約書

【要約】

【課題】 複数の通信装置が互いに動作要求及び受信した動作要求に対する動作応答を送受信する場合において、通信の効率を上げる。

【解決手段】 HTTPクライアント11が、HTTPサーバ12に送信すべきクライアントコマンドとHTTPサーバ12から受信したサーバコマンドに対する応答とを一括して1つのHTTPリクエストとしてHTTPサーバ12に送信し、HTTPサーバ12が、HTTPクライアントに送信すべきサーバコマンドとHTTPクライアント11から受信したクライアントコマンドに対する動作応答とを一括して1つのHTTPレスポンスとしてHTTPクライアント11に送信する。この場合において、各コマンドが関数呼び出しであり、これに対する応答が、その関数呼び出しによって呼び出された関数の実行結果であるとよい。

【選択図】

図 5

特願 2003-305513

出願人履歴情報

識別番号

[000006747]

1. 変更年月日

2002年 5月17日

[変更理由]

住所変更

住 所

東京都大田区中馬込1丁目3番6号

氏 名

株式会社リコー